CR 92.001

NCEL

Contract Report

March 1992

An Investigation Conducted by
B. Nour-Omid
Scopus Technology, Inc.

SOLVING LARGE LINEARIZED SYSTEMS IN MECHANICS

Abstract The solution of symmetric linearized systems of equations arising from applications of the finite element method to nonlinear analysis of structures is considered. Two related schemes that are based on Krylov sequences (e.g., Lanczos and conjugate gradient procedures) are examined. The conjugate gradient procedure is derived directly from the Lanczos process. In this derivation it is demonstrated that the conjugate gradient implicitly computes the triangular factors of the reduced tridiagonal system generated by the Lanczos process. The stability of the conjugate gradient procedure depends on that for the triangular factorization that can be guaranteed only for positive definite systems.

Loss of orthogonality among the Lanczos vectors is also addressed and the method of partial reorthogonalization is used to maintain orthogonality. This approach improves the robustness of the algorithm but can be expensive for ill-conditioned systems. For such problems, preconditioning can help reduce the number of iterations required for convergence. Three different preconditioning schemes are considered; diagonal, element-by-element (E x E), and substructure-by-substructure (S x S).

The above methods were applied to a number of different example problems. The S x S method is more effective than either the diagonal or the E x E methods at reducing both the computation cost and the number of iterations. For ill-conditioned systems the computation cost for E x E was less than that for diagonal preconditioners but as the condition number improved, the cost for the two methods were about the same.

NAVAL CIVIL ENGINEERING LABORATORY PORT HUENEME CALIFORNIA 93043-5003

METRIC CONVERSION FACTORS

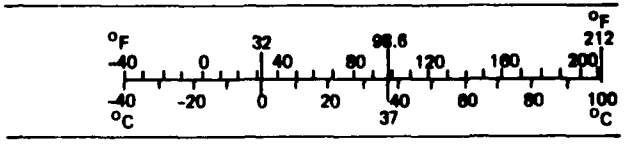
Approximate Conversions to Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
in	inches	*2.5	centimeters	cm
ft	feet	30	centimeters	cm
yd	yards	0.9	*meters*	*m*
mi	miles	1.6	kilometers	km
		AREA		
in^2	square inches	6.5	square centimeters	cm^2
ft^2	square feet	0.09	square meters	m^2
yd^2	square yards	0.8	square meters	m^2
mi^2	*square miles*	2.6	square kilometers	km^2
	acres	0.4	hectares	ha
		MASS (weight)		
oz	ounces	28	grams	g
lb	pounds	0.45	kilograms	kg
	short tons (2,000 lb)	0.9	tonnes	t
		VOLUME		
tsp	teaspoons	5	milliliters	ml
Tbsp	*tablespoons*	15	*milliliters*	*ml*
fl oz	fluid ounces	30	milliliters	ml
c	cups	0.24	liters	l
pt	pints	0.47	liters	l
qt	quarts	0.95	liters	l
gal	gallons	3.8	liters	l
ft^3	cubic feet	0.03	cubic meters	m^3
yd^3	cubic yards	0.76	cubic meters	m^3
		TEMPERATURE (exact)		
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

*1 in = 2.54 (exactly). For other exact conversions and more detailed tables, see NBS Misc. Publ. 286, Units of Weights and Measures, Price $2.25, SD Catalog No. C13.10:286.

Approximate Conversions from Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
mm	millimeters	0.04	inches	in
cm	centimeters	0.4	inches	in
m	meters	3.3	feet	ft
m	meters	1.1	yards	yd
km	kilometers	0.6	miles	mi
		AREA		
cm^2	square centimeters	0.16	square inches	in^2
m^2	square meters	1.2	square yards	yd^2
km^2	square kilometers	0.4	square miles	mi^2
ha	hectares (10,000 m^2)	2.5	acres	
		MASS (weight)		
g	*grams*	0.035	ounces	oz
kg	kilograms	2.2	pounds	lb
t	tonnes (1,000 kg)	1.1	short tons	
		VOLUME		
ml	milliliters	0.03	fluid ounces	fl oz
l	liters	2.1	pints	pt
l	liters	1.06	quarts	qt
l	liters	0.26	gallons	gal
m^3	cubic meters	35	cubic feet	ft^3
m^3	cubic meters	1.3	cubic yards	yd^3
		TEMPERATURE (exact)		
°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F



REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-018

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1992	Not Final; March 1991 - November 1991

4. TITLE AND SUBTITLE

SOLVING LARGE LINEARIZED SYSTEMS IN MECHANICS

5. FUNDING NUMBERS

PR - RM33F60-A2

6. AUTHOR(S)

B. Nour-Omid

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(S)

Scopus Technology, Inc.
1900 Powell Street, Suite 900
Emeryville, CA 94608

8. PERFORMING ORGANIZATION REPORT NUMBER

CR 92-001

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(S)

Office of Naval Technology / Naval Civil Engineering Laboratory
800 Quincy Street Code L51
Arlington, VA 22217-5000 Port Hueneme, CA 93043-5003

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The solution of symmetric linearized systems of equations arising from applications of the finite element method to nonlinear analysis of structures is considered. Two related schemes that are based on Krylov sequences (e.g., Lanczos and conjugate gradient procedures) are examined. The conjugate gradient procedure is derived directly from the Lanczos process. In this derivation it is demonstrated that the conjugate gradient implicitly computes the triangular factors of the reduced tridiagonal system generated by the Lanczos process. The stability of the conjugate gradient procedure depends on that for the triangular factorization that can be guaranteed only for positive definite systems.

Loss of orthogonality among the Lanczos vectors is also addressed and the method of partial reorthogonalization is used to maintain orthogonality. This approach improves the robustness of the algorithm but can be expensive for ill-conditioned systems. For such problems, preconditioning can help reduce the number of iterations required for convergence. Three different preconditioning schemes are considered; diagonal, element-by-element (E x E), and substructure-by-substructure (S x S).

The above methods were applied to a number of different example problems. The S x S method is more effective than either the diagonal or the E x E methods at reducing both the computation cost and the number of iterations. For ill-conditioned systems the computation cost for E x E was less than that for diagonal preconditioners but as the condition number improved, the cost for the two methods were about the same.

14. SUBJECT TERMS

Numerical methods, equation solving algorithms, linearized systems, finite element, Lanczos, conjugate gradient, preconditioning

15. NUMBER OF PAGES

36

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

1. Introduction

In this study we consider the solution of symmetric linear systems of equations arising from applications of the finite element method using Krylov based algorithms (e.g. Lanczos and conjugate gradient procedures). The Lanczos algorithm [1] was first introduced in 1950 as a method for computing eigenvalues and the corresponding eigenvectors of a matrix. In 1952 Hestenes and Stiefel [2] introduced the method of conjugate gradients (CG) for solving linear systems of equations. In the same year, Lanczos showed that his algorithm then called the method of minimized iteration [3], can also be used to obtain the solution of a linear system of equations. In fact, these methods are closely related in the sense that in exact arithmetic (when no roundoff errors are present) they compute the same approximate solution at each step. A fact known to both Lanczos and Hestenes.

An important motivating factor for using Lanczos and CG methods is the theoretical result showing that in exact arithmetic both methods are able to compute the solution in less than n iterations, where n is the number of equations in the system. In fact the number of iterations will be less than the total number of distinct eigenvalues in the system. In 1960, the CG method was first used to solve system of linear equations arising in structural mechanics [4]. In this paper, Lively showed that CG was not effective for solving the ill-conditioned systems that often arise in structural analysis. The popularity of the CG method vanished when it was found that for certain problems it required well over n steps to converge to the correct solution. CG was then abandoned and the more effective direct methods based on triangular factorization of the matrix were adopted by structural analysts as their method of choice.

Nonlinear transient finite element problems may be characterized by the equations of dynamic equilibrium

$$\mathbf{M}\ddot{\mathbf{w}} + \mathbf{f}^{int}(\mathbf{w}) = \mathbf{f}^{ext}(t) \tag{1}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{f}^{int}$ is the vector of internal resisting forces due to the displacements $\mathbf{w}$, and $\mathbf{f}^{ext}$ is the time dependent external force vector [5]. The above system is generally solved by applying a step-by-step time integration procedure resulting in a system of nonlinear algebraic equations. The solution to this system is obtained using a Newton-Raphson iteration or related schemes. At the heart of this iteration is a set of linear equations

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{2}$$

where $\mathbf{x}$ is the correction to the approximate solution vector in the nonlinear iteration loop. $\mathbf{A}$ is symmetric, positive definite and sparse and is related to the system in (1) through

$$\mathbf{A} = \mathbf{M} + \mathbf{K}\delta \tag{3}$$

where $\mathbf{K} = \frac{\partial \mathbf{f}^{int}}{\partial \mathbf{w}}$ and δ is a scaler parameter that depends on the time step (e.g. $\delta = \frac{1}{2}\Delta t^2$). For problems with small bandwidth or problems which result in small fill-in in the triangular factorization of

A, direct methods are the fastest solvers [6]. When the bandwidth of A is large (e.g. large complex two and three dimensional problems), the solution of the linear system may be a formidable task and alternative procedures must be considered.

In 1971 the advantages of Lanczos and CG methods were recognized when attention was focused on linear systems with large sparse matrix coefficients, see [7]. An important property of these methods is that, at each step, the solution to (2) can be obtained with out an explicit knowledge of the matrix. Only a means of computing the matrix vector product $\mathbf{Av}$ for a given vector $\mathbf{v}$ is required. This is an elegant way of exploiting the sparsity structure of A. Typically, in finite element analysis there are fewer than 100 nonzero terms in each row of A. The number of nonzero terms in A is independent of the number of equations. It depends on the number of nodes per element, the number of parameters per node, and the number of elements attached to a node. The number of equations in this system depends on the complexity of the structure (i.e. the domain) and also on the amount of detail desired in describing the solution.

Preconditioning is introduced to alleviated the difficulties associated with the slow convergence of Lanczos and CG methods. Instead of solving (2) one solves

$$\mathbf{AB}^{-1}\mathbf{y} = \mathbf{b} \tag{4}$$

where $\mathbf{x} = \mathbf{B}^{-1}\mathbf{y}$. B is referred to as the preconditioning matrix. The advantages of preconditioning can also be realized by solving $\mathbf{B}^{-1}\mathbf{Ax} = \mathbf{B}^{-1}\mathbf{b}$. The number of iterations required to solve (4) depends on the condition number, κ, of its coefficient matrix. $\kappa(\mathbf{AB}^{-1})$ is the ratio of the largest eigenvalue of the eigenproblem $[\mathbf{A} - \lambda\mathbf{B}]\mathbf{z} = \mathbf{0}$ to its the smallest (i.e $\kappa = \|\mathbf{AB}^{-1}\| \, \|\mathbf{BA}^{-1}\|$). Theoretical considerations suggest that at the end of each of the first few iterations of both Lanczos and CG methods the residual norm is reduced by a factor of $\dfrac{\sqrt{\kappa(\mathbf{AB}^{-1})}-1}{\sqrt{\kappa(\mathbf{AB}^{-1})}+1}$. Note that when κ is unity a single iteration is sufficient to solve the equation. Of course κ is one only when $\mathbf{B} = \mathbf{A}$! However, this provides us with a guideline for choosing B. B must be chosen such that one can easily compute the solution of a linear system of equations with B as its matrix coefficient while at the same time it is as close to A as possible. For non-trivial matrices A, these are contradictory requirements which makes the problem of finding good preconditioners a challenging one. For a well chosen B only a few iterations is required to reduce the residual norm to the desired level. It is important to note that the condition number of $\mathbf{AB}^{-1}$ depends on the time step through δ in equation (3) as the following example demonstrates.

Example 1:

Let $\mathbf{K} = \begin{bmatrix} 1+\zeta & -1 \\ -1 & 1+\zeta \end{bmatrix}$ and $\mathbf{M} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Then

$$\mathbf{A} = \begin{bmatrix} 1+(1+\zeta)\delta & -\delta \\ -\delta & 1+(1+\zeta)\delta \end{bmatrix}$$

The eigenvalues of the preconditioned system satisfies the quadratic equation $det[\mathbf{A} - \lambda\mathbf{B}] = 0$. There is no change in the condition number for this problem with diagonal preconditioning since the diagonal of **A** is a scalar multiple of the identity matrix. Then the condition number for this problem becomes

$$\kappa = \frac{1 + (2 + \zeta)\delta}{1 + \zeta\delta}$$

For a sufficiently small time step the condition number is close to unity. On the other hand as the time step increases the condition number approaches $1 + \frac{2}{\zeta}$ which may be arbitrarily large for small ζ.

This example demonstrates that; (a) the performance of iterative methods for solving linear systems of equations arising from transient finite element problems depends strongly on the time step, and (b) for a given finite element discreatization static problems result in worse conditioned system of equations than transient problems. These facts should be considered when assessing the performance of iterative methods.

Here, we focus on systems which arise from the application of the finite element method to engineering problems whose sparsity structure may be characterized by

$$\mathbf{A} = \sum_e \mathbf{N}_e \mathbf{a}_e \mathbf{N}_e^T \tag{5}$$

where $\mathbf{N}_e$ is long and thin Boolean connectivity matrix and $\mathbf{a}_e$ denotes the small stiffness matrix for element e. We can take advantage of this structure of **A** when using either CG or Lanczos methods to solve (2). In [8] and [9], it is pointed out that the matrix vector product

$$\mathbf{Au} = \sum_e (\mathbf{N}_e \mathbf{a}_e \mathbf{N}_e^T \mathbf{u}) \tag{6}$$

can be computed without ever assembling **A**. The evaluation of **Au** using (6) requires more arithmetic operations than that using an assembled **A** (assuming some compact structure where no zero entries of **A** are stored). Typically, the number of arithmetic operations would increase by about three folds. However, the use of parallel and vector computers produces only a modest increase in the elapsed time and in certain cases might even reduce it.

In 1983, Hughes, Levit and Winget [10] proposed a time integration algorithm for the solution of heat conduction equations that uses an element-by-element (E×E) splitting. In the same year, Ortiz, Pinsky and Taylor [11] proposed a novel extension of the E×E procedure to the solution of dynamic equations. These E×E time integration algorithms are unconditionally stable, but they lack accuracy which limits their use. Hughes, Levit and Winget [12] reformulate the E×E procedure as an iterative solver to achieve the accuracy and stability of standard finite element algorithms. In [13] Nour-Omid and Parlett addressed the problem of preconditioning (2). The idea is to employ the methods for solving differential equations presented in [10,11,12] as preconditioners. The resulting preconditioners use the element representation of **A** in (6), and requires no globally assembled matrix. They are defined as the product of positive definite element matrices obtained by applying a diagonal shift to the positive semi-definite element stiffness matrices. Winget and

Hughes [14] further developed the ideas of element preconditioners and constructed a variation that replaces the terms on the diagonals of the element matrices with the corresponding ones in the assembled matrix. This modification also results in positive definite element matrices. A product algorithm similar to that in [11] is then constructed using the Choleski factorization of these modified element matrices. It is worth noting that these preconditioners, though computed element-by-element, are an approximate Choleski factorization of **A**, see [15]. Our primary interest in element-by-element preconditioning is in keeping storage requirements down in the analysis of regular structures, but the advent of vector and parallel computing may make this approach a fast one as well, especially in three dimensions.

In section 2 we briefly describe the Lanczos algorithm and present a derivation of the conjugate gradient method from the Lanczos algorithm in section 3. We then turn to the problem of orthogonality loss that affects both methods. As a remedy, we consider the approach of partial reorthogonalization proposed by Simon [16]. The element preconditioners used in this study are described in section 5 together with a discussion of some implementation issues. In section 6 we describe the $S \times S$ preconditioners and its relation to $E \times E$ method. The results of numerical tests on two characteristically different problems are presented in section 7.

2. Lanczos Algorithm

When used as a method for solving linear systems, the Lanczos process starts from a given initial approximation to the solution, $\mathbf{x}_0$. Associated with $\mathbf{x}_0$, define the residual vector $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$. Unless a good estimate to the desired solution is available, the best choice for $\mathbf{x}_0$ is the zero vector. Then $\mathbf{r}_0 = \mathbf{b}$. Normalizing $\mathbf{r}_0$ gives the first Lanczos vector, $\mathbf{q}_1$. Implicitly, at the end of the first step the algorithm obtains a Galerkin approximation to the solution of (2) from the one dimensional space with $\mathbf{q}_1$ as the base vector. Associated with this approximation is a new residual vector. The normalization of this residual results in the second Lanczos vector, $\mathbf{q}_2$. Repeating the Galerkin process but using the two dimensional space, $span(\mathbf{q}_1, \mathbf{q}_2)$, followed by the normalization of the residual one obtains $\mathbf{q}_3$.

At a typical step, j, the Lanczos algorithm computes a residual vector associated with a best approximation, $\mathbf{x}_j$, (in a Galerkin sense) to $\mathbf{x}$ from the j dimensional space, $span[\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_j]$. This space is often referred to as the space of trial vectors. The next Lanczos vector, $\mathbf{q}_{j+1}$, is the normalized residual $\mathbf{b} - \mathbf{A}\mathbf{x}_j$. This process is repeated until the norm of the current residual is small compared to the that of the starting residual. The Galerkin method chooses the approximate solution $\mathbf{x}_j$ by forcing the associated residual to be orthogonal to the space of trial (Lanczos) vectors. The Lanczos method computes neither $\mathbf{x}_j$ nor the associated residual. Instead it computes, at step j, a j-vector $\mathbf{s}_j$ that contains the weighting parameters for constructing the Galerkin solution.

Alternatively, Lanczos may be described as the Gram-Schmit orthogonalization process applied to the Krylov space, $[\mathbf{r}_0, \mathbf{A}\mathbf{B}^{-1}\mathbf{r}_0, (\mathbf{A}\mathbf{B}^{-1})^2\mathbf{r}_0, ..., (\mathbf{A}\mathbf{B}^{-1})^{j-1}\mathbf{r}_0]$, associated with equation (4). The orthogonalization is performed with respect to the $\mathbf{B}^{-1}$ inner product. The result of this orthogonalization is the set of Lanczos vectors $[\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_j]$. $\mathbf{q}_{j+1}$ is obtained by orthonormalizing $(\mathbf{A}\mathbf{B}^{-1})^j\mathbf{r}_0$ against the computed Lanczos vectors. The same vector $\mathbf{q}_{j+1}$ is obtained if $\mathbf{A}\mathbf{B}^{-1}\mathbf{q}_j$ is used instead of $(\mathbf{A}\mathbf{B}^{-1})^j\mathbf{r}_0$. It turns out that the components of $\mathbf{A}\mathbf{B}^{-1}\mathbf{q}_j$ along the first $j-2$ Lanczos vectors are zero and orthogonalization needs to be performed only against $\mathbf{q}_j$ and $\mathbf{q}_{j-1}$. The result is a vector $\mathbf{r}_j$ in the same direction as the residual due to the Galerkin approximation described above.

The algorithm can then be rewritten as the three term relation

$$\mathbf{r}_j = \beta_{j+1}\mathbf{q}_{j+1} = \mathbf{A}\mathbf{B}^{-1}\mathbf{q}_j - \mathbf{q}_j\alpha_j - \mathbf{q}_{j-1}\beta_j \tag{7}$$

where $\alpha_j = \mathbf{q}_j^T\mathbf{B}^{-1}\mathbf{A}\mathbf{B}^{-1}\mathbf{q}_j$ and $\mathbf{r}_j$ is normalized with respect to the inverse of the preconditioner to obtain $\mathbf{q}_{j+1}$ with normalizing factor $\beta_{j+1} = (\mathbf{r}_j^T\mathbf{B}^{-1}\mathbf{r}_j)^{\frac{1}{2}}$.

The j-th step of the Lanczos algorithm involves the calculation of α_j, β_{j+1}, and $\mathbf{q}_{j+1}$, in that order. In addition to the storage needs for $\mathbf{A}$ and $\mathbf{B}$, the algorithm requires storage for 5 vectors of length n; one for each of the vectors, $\mathbf{q}_{j-1}$, $\mathbf{q}_j$, $\mathbf{r}_j$, $\mathbf{p}_j = \mathbf{B}^{-1}\mathbf{q}_j$ and $\mathbf{p}_{j-1}$. The total cost for one step of the algorithm involves one solve with the preconditioner $\mathbf{B}$ as the coefficient matrix, a multiplication of $\mathbf{A}$ by a vector, two inner products and four products of a scalar by a vector. A summary of the Lanczos algorithm is presented in Table 1.

After m Lanczos steps all the quantities obtained from equation (7) can be arranged in a global matrix form

$$\left[\begin{array}{c|c} \mathbf{AB}^{-1} & \end{array}\right]\left[\begin{array}{c|c} \mathbf{Q}_m & \end{array}\right] - \left[\begin{array}{c|c} \mathbf{Q}_m & \end{array}\right][\mathbf{T}_m] = \left[\begin{array}{c|c} \mathbf{0} & \mathbf{r}_m \end{array}\right] = \mathbf{r}_m \mathbf{e}_m^T \tag{8}$$

Here $\mathbf{e}_m^T = (0, 0, ..., 0, 1)$, $\mathbf{Q}_m$ is an $n \times m$ matrix with columns $\mathbf{q}_i$, $i = 1, 2, ..., m$, and $\mathbf{T}_m$ is the tridiagonal matrix

$$\mathbf{T}_m = \begin{bmatrix} \alpha_1 & \beta_2 & & & & & \\ \beta_2 & \alpha_2 & \beta_3 & & & & \\ & \beta_3 & \cdot & \cdot & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & \beta_m \\ & & & & & \beta_m & \alpha_m \end{bmatrix} \tag{9}$$

The orthogonality property of the Lanczos vectors, $\mathbf{Q}_m^T \mathbf{B}^{-1} \mathbf{Q}_m = \mathbf{I}_m$, where $\mathbf{I}_m$ is the $m \times m$ identity matrix, can be used in equation (8) to obtain

$$\mathbf{Q}_m^T \mathbf{B}^{-1} \mathbf{A} \mathbf{B}^{-1} \mathbf{Q}_m = \mathbf{T}_m \tag{10}$$

A Galerkin approximation to $\mathbf{y}$ in (4) can be constructed by taking a linear combination of the Lanczos vectors. Accordingly,

$$\mathbf{y}_m = \mathbf{Q}_m \mathbf{s}_m \tag{11}$$

where $\mathbf{s}_m$ satisfies the tridiagonal system of equations

$$\mathbf{T}_m \mathbf{s}_m = \mathbf{Q}_m^T \mathbf{B}^{-1} \mathbf{r}_0 = \beta_1 \mathbf{e}_1 \tag{12}$$

The last equality is obtained using the fact that the starting vector is $\mathbf{r}_0 = \beta_1 \mathbf{q}_1$. $\mathbf{e}_1$ is the first column of the identity matrix. Equation (12) is a weak form of (4) and is obtained by first substituting the approximation to $\mathbf{y}$ from (11) into equation (4) to obtain the residual

$$\mathbf{g}_m = \mathbf{A} \mathbf{B}^{-1} \mathbf{Q}_m \mathbf{s}_m - \mathbf{b} \tag{13}$$

Orthogonalizing $\mathbf{g}_m$ against $\mathbf{Q}_m$ with respect to the $\mathbf{B}^{-1}$ inner product results in equation (12). $\mathbf{g}_m$ is simply related to $\mathbf{r}_m$ through

$$\mathbf{g}_m = \mathbf{r}_m \sigma_m \tag{14}$$

where σ_m is the bottom element of $\mathbf{s}_m$. The norm of this residual, $\rho_m = \|\mathbf{g}_m\| = \beta_{m+1} |\sigma_m|$, can be used to monitor the convergence. Once ρ_m is sufficiently small the Lanczos algorithm is terminated and the solution is constructed using (11). The Lanczos vectors can be put on secondary storage as they are being generated. There are two main reasons for keeping the Lanczos vectors.

(a) They are used occasionally in subsequent steps to restore orthogonality (see the following section on Loss of Orthogonality for more details).

(b) They can be recalled and used to construct the solution to a new right hand side [17]. The algorithm in Table 1 is particularly well suited for multiple right hand sides.

Given an approximate solution vector $\mathbf{x}_0$:

(1) Set

(a) $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$,

(b) $\mathbf{q}_0 = \mathbf{0}$,

(c) Solve $\mathbf{B}\bar{\mathbf{p}}_1 = \mathbf{r}_0$.

(d) $\beta_1 = (\mathbf{p}_1^T \mathbf{r}_0)^{\frac{1}{2}}$,

(e) $\mathbf{q}_1 = \frac{1}{\beta_1}\mathbf{r}_0$

(f) $\mathbf{p}_1 = \frac{1}{\beta_1}\bar{\mathbf{p}}_1$

(2) for $j = 1, 2, \ldots$ repeat;

(a) $\bar{\mathbf{r}}_j = \mathbf{A}\mathbf{p}_j$

(b) $\hat{\mathbf{r}}_j = \bar{\mathbf{r}}_j - \mathbf{q}_{j-1}\beta_j$

(c) $\alpha_j = \mathbf{q}_j^T \mathbf{B}^{-1}\hat{\mathbf{r}}_j = \mathbf{p}_j^T \hat{\mathbf{r}}_j$

(d) $\mathbf{r}_j = \hat{\mathbf{r}}_j - \mathbf{q}_j \alpha_j$

(e) Solve $\mathbf{B}\bar{\mathbf{p}}_j = \mathbf{r}_j$

(f) $\beta_{j+1} = (\mathbf{r}_j^T \mathbf{B}^{-1}\mathbf{r}_j)^{\frac{1}{2}} = (\bar{\mathbf{p}}_j^T \mathbf{r}_j)^{\frac{1}{2}}$

(g) if residual norm is small then terminate the loop.

(h) $\mathbf{q}_{j+1} = \frac{1}{\beta_{j+1}}\mathbf{r}_j$

(i) $\mathbf{p}_{j+1} = \frac{1}{\beta_{j+1}}\bar{\mathbf{p}}_j$

(3) Solution $\mathbf{x} = \mathbf{x}_0 + \mathbf{B}^{-1}\mathbf{Q}_m \mathbf{s}_m$

Table 1. The Lanczos algorithm.

3. Conjugate Gradient Algorithm

In this section we give a derivation of the conjugate gradient algorithm directly from the Lanczos process [18,19,20]. The conjugate gradient method can be viewed as a procedure that implicitly computes the triangular factorization of $\mathbf{T}_m$ through an update algorithm to combine the steps 2 and 3 of the Lanczos algorithm given in table 1. Accordingly

$$\mathbf{T}_m = \mathbf{L}_m \mathbf{D}_m \mathbf{L}_m^T \tag{15}$$

where $\mathbf{D}_m = diag[\delta_1, \delta_2, \ldots, \delta_m]$ and

$$\mathbf{L}_m = \begin{bmatrix} 1 & & & & & \\ -\omega_1 & 1 & & & & \\ & -\omega_2 & 1 & & & \\ & & \cdot & & & \\ & & & \cdot & & \\ & & & \cdot & 1 & \\ & & & & -\omega_{m-1} & 1 \end{bmatrix} \tag{16}$$

The components of $\mathbf{T}_m$, $\mathbf{L}_m$, and $\mathbf{D}_m$ are related through the following pair of equations.

$$\begin{aligned} \delta_k &= \alpha_k - \omega_{k-1}^2 \delta_{k-1} \\ \omega_k &= -\frac{\beta_{k-1}}{\delta_k} \end{aligned} \tag{17}$$

These two equations completely define the algorithm for triangular factorization of $\mathbf{T}_m$. Next we define

$$\mathbf{Z}_m = \mathbf{B}^{-1} \mathbf{Q}_m \mathbf{L}_m^{-T} \tag{18}$$

An important property of $\mathbf{Z}_m$ is that its columns are orthogonal with respect to $\mathbf{A}$. To show this consider

$$\begin{aligned} \mathbf{Z}_m^T \mathbf{A} \mathbf{Z}_m &= \mathbf{L}_m^{-1} \mathbf{Q}_m \mathbf{B}^{-1} \mathbf{A} \mathbf{B}^{-1} \mathbf{Q}_m \mathbf{L}_m^{-T} \\ &= \mathbf{L}_m^{-1} \mathbf{T}_m \mathbf{L}_m^{-T} \\ &= \mathbf{D}_m \end{aligned}$$

The columns of $\mathbf{Z}_m$ are said to be conjugate and the orthogonality condition of $\mathbf{Z}_m$ with respect to $\mathbf{A}$ is referred to as the conjugacy condition.

Multiplying both sides of equation (18) by $\mathbf{L}_m^T$ and using the bi-diagonal structure of $\mathbf{L}_m$ to equate the k-th column on either side of this equation, yields

$$\mathbf{z}_k - \mathbf{z}_{k-1} \omega_{k-1} = \mathbf{B}^{-1} \mathbf{q}_k \tag{19}$$

Defining $\mathbf{d}_k = \mathbf{B}^{-1} \mathbf{q}_k$ the above equation reduces to

$$\mathbf{z}_k = \mathbf{d}_k + \mathbf{z}_{k-1} \omega_{k-1} \tag{20}$$

Due to the conjugacy property of $\mathbf{Z}_m$ we are able to update the solution vector $\mathbf{x}_k$ by simply adding a component of $\mathbf{z}_k$. Thus, using equation (4) we have

$$\begin{aligned}
\mathbf{x}_k &= \mathbf{B}^{-1}\mathbf{y}_k \\
&= \mathbf{B}^{-1}\mathbf{Q}_k\mathbf{T}_k^{-1}\beta_1\mathbf{e}_1 \\
&= \mathbf{B}^{-1}\mathbf{Q}_k\mathbf{L}_k^{-T}\mathbf{D}_k^{-1}\mathbf{L}_k^{-1}\beta_1\mathbf{e}_1 \\
&= \mathbf{Z}_k\mathbf{D}_k^{-1}\mathbf{L}_k^{-1}\beta_1\mathbf{e}_1 \\
&= \mathbf{x}_{k-1} + \gamma_k\mathbf{z}_k
\end{aligned}$$

where γ_k is the k-th element of $\mathbf{D}_m^{-1}\mathbf{L}_m^{-1}\beta_1\mathbf{e}_1$. This way the residual vector can also be updated using the update relation

$$\mathbf{g}_k = \mathbf{g}_{k-1} - \gamma_k\mathbf{u}_k \tag{21}$$

and $\mathbf{u}_k = \mathbf{A}\mathbf{z}_k$. γ_k is related to the component of $\mathbf{D}_m$ and $\mathbf{L}_m$ through

$$\gamma_k = \frac{\rho_k}{\delta_k} \tag{22}$$

where ρ_k is updated through

$$\rho_k = \omega_{k-1}\rho_{k-1} \tag{23}$$

with $\rho_1 = \beta_1$. The above equation is simply the forward reduction algorithm to compute $\mathbf{L}_m^{-1}\beta_1\mathbf{e}_1$.

Thus the CG method directly computes the triangular factors of $\mathbf{T}_m$ by updating the factors of $\mathbf{T}_{m-1}$. The result is the algorithm in Table 2. It is important to note that $\mathbf{T}_m$ is often indefinite when $\mathbf{A}$ is a symmetric indefinite matrix. In this case the conjugate gradient algorithm is not reliable since the triangular factorization of $\mathbf{T}_m$ may be numerically unstable. This instability occurs when ever δ_k is small. Note that $\delta_k = \mathbf{z}_k^T\mathbf{u}_k$ is the denominator of the right hand side of (2b) in Table 2.

The CG algorithm generates a sequence of approximations, $\mathbf{x}_k$, to the solution $\mathbf{x}$ with a corresponding residual vector $\mathbf{g}_k$. The termination criterion can be chosen based on these quantities. In addition to storage demands for $\mathbf{A}$ and $\mathbf{B}$ the algorithm requires storage for 4 vectors.

Given an approximate solution $\mathbf{x}_0$ then:

(1) Set

(a) $\mathbf{g}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$

(b) $\mathbf{z}_1 = \mathbf{g}_0$

(c) Solve $\mathbf{B}\mathbf{d}_1 = \mathbf{g}_0$

(d) $\rho_1 = \mathbf{g}_0^T \mathbf{d}_1$

(2) for $k = 1, 2, \ldots$ repeat;

(a) $\mathbf{u}_k = \mathbf{A}\mathbf{z}_k$

(b) $\gamma_k = \frac{\rho_k}{\mathbf{z}_k^T \mathbf{u}_k}$

(c) $\mathbf{x}_k = \mathbf{x}_{k-1} + \gamma_k \mathbf{z}_k$

(d) $\mathbf{g}_k = \mathbf{g}_{k-1} - \gamma_k \mathbf{u}_k$

(e) Solve $\mathbf{B}\mathbf{d}_{k+1} = \mathbf{g}_k$.

(f) $\rho_{k+1} = \mathbf{g}_k^T \mathbf{d}_{k+1}$

(g) if $\rho_{k+1} \leq tol \cdot \rho_0$ then terminate the loop.

(h) $\omega_k = \frac{\rho_{k+1}}{\rho_k}$

(i) $\mathbf{z}_{k+1} = \mathbf{d}_{k+1} + \omega_k \mathbf{z}_k$

Table 2. The Conjugate Gradient Algorithm

4. Loss of Orthogonality

In finite precision, each computation introduces a small error and therefore the computed quantities will differ from their exact counterparts. Our objective here is to state the effect of roundoff error on the Lanczos process. For this purpose we denote by ε the smallest number in the computer such that $1 + \varepsilon > 1$. It is known as the unit roundoff error.

Although the tridiagonal relation, Eq. (8), is preserved to within roundoff, the $\mathbf{B}^{-1}$ orthogonality property of the Lanczos vectors completely breaks down after a certain number of steps depending on ε and the distribution of the eigenvalues of $\mathbf{B}^{-1}\mathbf{A}$ [16,19]. The Lanczos vectors not only lose their orthogonality, but may even become linearly dependent. This problem also effects the conjugate gradient method in the form of loss of conjugacy. A direct consequence of this loss of orthogonality is delay in convergence to the desired solution.

The loss of orthogonality can be viewed as the subsequent amplification of the errors introduced after each computation. We let $\mathbf{Q}_m$ denote the computed Lanczos vectors and define the following matrix

$$\mathbf{H}_m = \mathbf{Q}_m^T \mathbf{B}^{-1} \mathbf{Q}_m \tag{24}$$

In exact arithmetic $\mathbf{H}_m$ is the identity matrix. The off-diagonals of $\mathbf{H}_m$ will depend on ε, the unit roundoff error. Simon [16] found a recurrence relation that can be used to estimate the elements of a column of $\mathbf{H}_m$ from the elements of $\mathbf{T}_m$ and the elements in the previous columns of $\mathbf{H}_m$. This recursion can be stated in vector form

$$\beta_{j+1}\mathbf{h}_{j+1} \approx \mathbf{T}_{j-1}\mathbf{h}_j - \alpha_j \mathbf{h}_j - \beta_j \mathbf{h}_{j-1} \tag{25}$$

where $\mathbf{h}_{j-1}$, $\mathbf{h}_j$ and $\mathbf{h}_{j+1}$ are vectors of length $j-1$ containing the top $j-1$ elements of the $j-1$, j, and $j+1$-th columns of $(\mathbf{H}_m - \mathbf{I}_m)$. Here, the bottom element of $\mathbf{h}_{j-1}$ is ε. The orthogonality state can be monitored by updating $\mathbf{h}_{j+1}$ in the course of the Lanczos algorithm.

A number of preventive measures can be taken to maintain a certain level of orthogonality. Lanczos was aware of the effects of roundoff on the algorithm when he presented his work. He proposed that the newly computed vector, $\mathbf{q}_{j+1}$, be explicitly orthogonalized against all the preceding vectors at the end of each step. We will refer to this technique as "full reorthogonalization" method. It enforces orthogonality to within roundoff (i.e. $|\mathbf{q}_i^T \mathbf{B}^{-1} \mathbf{q}_j| < n\varepsilon, i \neq j$). In [16] Simon showed that the computed tridiagonal remains accurate to within roundoff if the more relaxed orthogonality condition

$$|\mathbf{q}_i^T \mathbf{B}^{-1} \mathbf{q}_j| < \sqrt{n\varepsilon}, i \neq j \tag{26}$$

is enforced. We refer to this as the semi-orthogonality condition, and to procedures that adopt the weaker condition as selective orthogonalization methods. Simon proposed to update $\mathbf{h}_{j+1}$ using (26) and monitor the magnitude of its elements. Whenever any component of $\mathbf{h}_{j+1}$ is greater than $\sqrt{\varepsilon}$ then semi-orthogonality

may be lost between $\mathbf{q}_{j+1}$ and some columns of $\mathbf{Q}_j$. At this step the appropriate Lanczos vectors are brought in from secondary store and $\mathbf{q}_{j+1}$ is orthogonalized against each of them. This operations must be carried out in two successive steps to avoid propagation of the errors.

A variant of Simon's scheme is to restore orthogonality of $\mathbf{q}_j$ and $\mathbf{q}_{j+1}$ at the same time. In this way no reorthogonalization of $\mathbf{q}_{j+2}$ will be necessary, at the end of the next step. The number of operations for this scheme is the same as that of the scheme above, but vectors are retrieved only once and therefore the I/O overhead is halved.

The disadvantage of reorthogonalization is that additional storage is required to keep the Lanczos vector. If m steps are required to reduce the residual to the desired level then storage for m vectors of length n is needed. The advantage is that reorthogonalization can significantly reduce the number of steps. This is demonstrated by the numerical examples.

5. Element by Element Preconditioning

The idea of using solution algorithms from discretized partial differential equations for constructing preconditioners is not new. As early as 1963 Wachspress proposed one such preconditioner based on the ADI method [21]. Recently, in [13] we proposed a number of preconditioners, based on the element-by-element representation of **A** for solving $\mathbf{M}\dot{\mathbf{x}} + \mathbf{A}\mathbf{x} = \mathbf{b}$. Here **M** is a diagonal matrix. The steady state solution of this equation is the same as that of (2).

The first preconditioner uses a Choleski factorization of the element matrices shifted by a diagonal matrix. We denote the diagonally scaled **A** by

$$\bar{\mathbf{A}} = \mathbf{M}^{-\frac{1}{2}}\mathbf{A}\mathbf{M}^{-\frac{1}{2}} = \sum_{e=1}^{n_e} \mathbf{N}_e \bar{\mathbf{a}}_e \mathbf{N}_e^T \tag{27}$$

where, n_e is the total number of elements. Then the proposed preconditioner can be constructed in the following manner. First, the Choleski factors $\bar{\mathbf{c}}_e\bar{\mathbf{c}}_e^T = \sigma\mathbf{I} + \bar{\mathbf{a}}_e$ is computed. The shift was applied to eliminate the singularity of $\bar{\mathbf{a}}_e$. A lower triangular matrix, $\bar{\mathbf{C}}$ is formed as the product of the $\bar{\mathbf{c}}_e$'s. $\bar{\mathbf{C}}$ is an approximation to the Choleski factorization of $\bar{\mathbf{A}}$. The resulting preconditioner is given by

$$\mathbf{B}^{-1} = \mathbf{M}^{\frac{1}{2}} \prod_{e=1}^{n_e} \mathbf{N}_e \bar{\mathbf{c}}_e^{-1} \mathbf{N}_e^T \prod_{e=n_e}^{1} \mathbf{N}_e \bar{\mathbf{c}}_e^{-T} \mathbf{N}_e^T \mathbf{M}^{\frac{1}{2}} \tag{28}$$

Note that the second product is carried out in the reverse order of the first. Numerical results indicated that a shift $\sigma = 1$ results in a preconditioner that is close to the optimum.

Writing $\bar{\mathbf{a}}_e = \bar{\mathbf{u}}_e + \bar{\mathbf{u}}_e^T + \bar{\mathbf{d}}_e$, where $\bar{\mathbf{d}}_e$ and $\bar{\mathbf{u}}_e$ denote the diagonal and strict upper triangular part of $\bar{\mathbf{a}}_e$, a second preconditioner was constructed as

$$\mathbf{B}^{-1} = \mathbf{M}^{\frac{1}{2}} \prod_{e=1}^{n_e} \mathbf{N}_e (\mathbf{I} + \bar{\mathbf{d}}_e + \bar{\mathbf{u}}_e^T)^{-1} \mathbf{N}_e^T \prod_{e=n_e}^{1} \mathbf{N}_e (\mathbf{I} + \bar{\mathbf{d}}_e + \bar{\mathbf{u}}_e)^{-1} \mathbf{N}_e^T \mathbf{M}^{\frac{1}{2}} \tag{29}$$

A comparison of these two preconditioners on small problems indicated that the Choleski form is more effective.

E×E Choleski:

In [14] Winget replaced the diagonal of $\bar{\mathbf{a}}_e$ with the identity matrix to form

$$\mathbf{B}^{-1} = \mathbf{M}^{\frac{1}{2}} \prod_{e=1}^{n_e} \mathbf{N}_e \tilde{\mathbf{c}}_e^{-1} \mathbf{N}_e^T \prod_{e=n_e}^{1} \mathbf{N}_e \tilde{\mathbf{c}}_e^{-T} \mathbf{N}_e^T \mathbf{M}^{\frac{1}{2}} \tag{30}$$

where $\tilde{\mathbf{c}}_e\tilde{\mathbf{c}}_e^T = \mathbf{I} + \bar{\mathbf{u}}_e + \bar{\mathbf{u}}_e^T$. This avoids the artificial shifting of $\bar{\mathbf{a}}_e$.

E×E LU Split:

Similarly, by dropping $\bar{\mathbf{d}}_e$ in (29), a new but simpler preconditioner

$$\mathbf{B} = \mathbf{M}^{\frac{1}{2}} \prod_{e=1}^{n_e} \mathbf{N}_e(\mathbf{I} + \bar{\mathbf{u}}_e^T)\mathbf{N}_e^T \prod_{e=n_e}^{1} \mathbf{N}_e(\mathbf{I} + \bar{\mathbf{u}}_e)\mathbf{N}_e^T\mathbf{M}^{\frac{1}{2}} \tag{31}$$

can be constructed that eliminates the need for forming Choleski factorization of element matrices. The main advantage is the reduction in storage since $\mathbf{B}^{-1}\mathbf{v}$ can be evaluated for any $\mathbf{v}$ using the element representation of $\mathbf{A}$. In the next section we compare the two preconditioners defined in (30) and (31) using both Lanczos and CG methods.

Implementation

We view the computation of the matrix-vector product $\mathbf{Au}$ via equation (6) as a mechanism for saving storage in return for extra arithmetic work. The reduction in storage demand is due to the following:

1. In most practical finite element problems there is a considerable amount of repetition of a given element in the mesh structure.
2. The element matrices of a number of element types, such as beams, trusses, etc., are known explicitly and depend on only a few fundamental parameters.

The first observation allows us to create a data structure which keeps the element matrix of one element to represent a whole group of elements. The second observation results in a canonical form for each element type, and therefore only a few parameters need be stored to define each element matrix. Hence the storage requirements for all the distinct $\mathbf{a}_e$ is often significantly less than the number of words required to hold $\mathbf{A}$, even when a sophisticated sparse storage scheme is used (see [6]). Furthermore, one can always recompute the element matrices $\mathbf{a}_e$ each time the product $\mathbf{Au}$ is required.

The overhead for the reduction in storage is the increased number of operations. However, two comments are in order:

1. The cost of a multiply no longer dominates arithmetic evaluations.
2. Vector and Parallel computers or other special purpose devices can execute $\sum_e(\mathbf{N}_e\mathbf{a}_e\mathbf{N}_e^T\mathbf{u})$ very efficiently.

When using the implicit form of $\mathbf{Au}$ it can be seen that different elements operate on different parts of the vector $\mathbf{v}$. One can take advantage of this fact by performing some of the element matrix operations in parallel. The elements are simply arranged into p groups, where p is the number of available processors. Each processor then computes the contribution of the product of all the element matrices in its assigned group by the corresponding components of $\mathbf{v}$. Finally, the contribution from each group is combined to obtain $\mathbf{Au}$. The implicit product increases the cost of matrix operations by a factor of, say μ, where μ depends on the average number of elements connected to a node. Typically μ range between 1.5 and 3 [13], although one could design examples that result in large μ.

6. Substructure by Substructure Preconditioning

An obvious generalization of the element by element preconditioner described above is the substructure by substructure (S×S) preconditioner. Here the finite element mesh is partitioned into a number of substructures (also referred to as sub-domains or super-elements). Each substructure consists of a group of elements. Associated with each substructure one can define a stiffness matrix. The assembly of the substructure stiffness matrices results in the global matrix

$$\mathbf{A} = \sum_{S} \mathbf{N}_S \mathbf{A}_S \mathbf{N}_S^T \tag{32}$$

where $\mathbf{A}_S$ denotes the stiffness matrix for a substructure. This is a generalization of the element by element preconditioner in the sense that for the special case when each substructure consists of a single element, equation (32) reduce to (5).

Following a similar approach to the development of E×E preconditioner we are required to perform some form of factorization with $\mathbf{A}_S$. It is important to note that any preconditioner obtained from the factorization $\mathbf{A}_S$ will depend on the ordering of the unknowns associated with the parameters in the given substructure. By adopting a special ordering where the interior nodes for each substructure are numbered first one can arrive at the hybrid scheme proposed in [23]. Then, the unknowns associated with each substructure can be partitioned as

$$\mathbf{x}_S = \begin{Bmatrix} \mathbf{x}_S^I \\ \mathbf{x}_S^B \end{Bmatrix} \tag{33}$$

where $\mathbf{x}_S^I$ denotes the unknowns in the *interior*, and $\mathbf{x}_S^B$ denotes the unknowns on the *boundary*. Here, all quantities associated with boundary and interior unknowns are denoted with superscripts B and I, respectively. The terms in equation (32) can be expressed as

$$\mathbf{A}_S = \begin{bmatrix} \mathbf{A}_S^{II} & \mathbf{A}_S^{IB} \\ \mathbf{A}_S^{IB\,T} & \mathbf{A}_S^{BB} \end{bmatrix} \tag{34}$$

and

$$\mathbf{N}_S = [\mathbf{0}, \mathbf{0} \ldots \mathbf{I}, \mathbf{0} \ldots \mathbf{0}, \mathbf{N}_S^B] \tag{35}$$

The first part of $\mathbf{N}_S$ consists of zero blocks except for an identity block corresponding to the interior unknowns. The right hand side vector may also be partitioned in a similar manner to obtain

$$\mathbf{b}_S = \begin{Bmatrix} \mathbf{b}_S^I \\ \mathbf{b}_S^B \end{Bmatrix} \tag{36}$$

With the above ordering the linear system in (2) takes the form

$$\begin{bmatrix} \mathbf{A}_1^{II} & & & & \mathbf{A}_1^{IB}\mathbf{N}_1^{B\,T} \\ & \mathbf{A}_2^{II} & & & \mathbf{A}_2^{IB}\mathbf{N}_1^{B\,T} \\ & & \ddots & & \vdots \\ & & & \mathbf{A}_S^{II} & \mathbf{A}_S^{IB}\mathbf{N}_1^{B\,T} \\ \mathbf{N}_1^B\mathbf{A}_1^{IB\,T} & \mathbf{N}_2^B\mathbf{A}_1^{IB\,T} & \ldots & \mathbf{N}_S^B\mathbf{A}_S^{IB\,T} & \mathbf{A}^{BB} \end{bmatrix} \begin{Bmatrix} \mathbf{x}_1^I \\ \mathbf{x}_2^I \\ \vdots \\ \mathbf{x}_S^I \\ \mathbf{x}^B \end{Bmatrix} = \begin{Bmatrix} \mathbf{b}_1^I \\ \mathbf{b}_2^I \\ \vdots \\ \mathbf{b}_S^I \\ \mathbf{b}^B \end{Bmatrix} \tag{37}$$

where $\mathbf{A}^{BB}$, $\mathbf{b}^B$, and $\mathbf{x}^B$ are related to quantities in equations (33), (34), (35) and (36) through

$$\begin{aligned} \mathbf{A}^{BB} &= \sum_S \mathbf{N}_S^B \mathbf{A}_S^{BB} \mathbf{N}_S^{B^T} \\ \mathbf{b}^B &= \sum_S \mathbf{N}_S^B \mathbf{b}_S^B \\ \mathbf{x}^B &= \mathbf{N}_S^{B^T} \mathbf{x}^B \end{aligned} \tag{38}$$

The block arrow structure of the coefficient matrix in (37) is due to the fact that the interior unknowns in one substructure interact with those in a second substructure only through the boundary unknowns $\mathbf{x}^B$. Eliminating the interior unknowns and using the definitions in (38) one obtains the linear system of equations for the boundary unknows

$$\left[\sum_S \mathbf{N}_S^B \bar{\mathbf{A}}_S^{BB} \mathbf{N}_S^{B^T}\right] \mathbf{x}^B = \sum_S \mathbf{N}_S^B \bar{\mathbf{b}}_S^B \tag{39}$$

where $\bar{\mathbf{b}}_S^B = \mathbf{b}_S^B - \mathbf{A}_S^{IB^T} (\mathbf{A}_S^{II})^{-1} \mathbf{b}_S^I$ and $\bar{\mathbf{A}}_S^{BB} = \mathbf{A}_S^{BB} - \mathbf{A}_S^{IB^T} (\mathbf{A}_S^{II})^{-1} \mathbf{A}_S^{IB}$ is the Schur complement of $\mathbf{A}_S^{II}$ in (34). The main advantage with this approach is that the matrix in (39) has a better condition number that the original system (see [23] for detail). It is important to note that these quantities can be computed independently of the other substructures and therefore completely in parallel.

The structure of matrix coefficient in equation (39) is similar to the matrix in (5) in the sense that they are both constructed using an assembly process. The only difference is that in (39) one is dealing with larger matrices. This similarity may be used to construct preconditioners for equation (39) in much the same way as in section 5. However, the difficulty with the product form for the preconditioners defined in section (5) is the sequential nature of the algorithm. In general, the product form in equations (30) and (31) when applied to the S×S partition requires the processing of substructure one at a time. This can hinder parallelism when implemented on a multi-processor computers. Although there are schemes designed to minimize the impact of the product form on parallel implementation (through graph coloring algorithms), when the number of substructures are close to the number of available processors these schemes are not effective.

An alternative preconditioner that is suitable for concurrent implementation can be constructed using the splitting algorithm proposed in [24-26] for transient finite element analysis. This scheme when applied to the problem in (1) results in an algorithm that has an additive form and thus lends itself to parallel implementation. Thus, the new preconditioner takes the form

$$\mathbf{B}^{-1} = \left[\sum_S \mathbf{N}_S^B \mathbf{U}_S^{\bar{B}B^{-T}} \mathbf{N}_S^{B^T}\right] \bar{\mathbf{D}} \left[\sum_S \mathbf{N}_S^{B^T} \mathbf{U}_S^{\bar{B}B^{-1}} \mathbf{N}_S^B\right] \tag{31}$$

where $\bar{\mathbf{D}}$ is diagonal matrices and $\mathbf{U}_S^{BB}$ is the lower triangle of $\bar{\mathbf{A}}_S^{BB}$. A good choice for $\bar{\mathbf{D}}$ is the diagonal of the coefficient matrix in (39) obtain by simply assembling the diagonals of $\bar{\mathbf{A}}_S^{BB}$.

The steps in evaluating $\mathbf{B}^{-1}\mathbf{v}$ for a given vector $\mathbf{v}$ is similar to those for computing the product pf $\bar{\mathbf{A}}_S^{BB}$ and a vector. First, $\mathbf{v}$ is localized to each substructure through $\mathbf{N}_S^B \mathbf{v}$. This is followed by a step of forward

reduction using the lower part of $\bar{\mathbf{A}}_S^{BB}$. Note that the forward reduction is performed for each substructure independently of the others. The result is then assembled to obtain a new vector which is then multiplied by $\bar{\mathbf{D}}$. A second localization of last result followed by a back-substitution and assembly completes the operation with the preconditioners.

7. Numerical Examples

In this section we discuss the results obtained from the application of the Lanczos and CG methods to two different example problems. The first example is a cavity driven flow problem (Stokes flow). The incompressibility of the fluid is represented by local volumetric constraints. These constraints are enforced in each finite element using a penalty method. The penalty parameter represents the bulk modulus of the fluid. 400 elements are used to model this problem. See figure 1(a). The condition number of A increases with the penalty parameter. We refer to this as material ill-conditioning.

The second example we used is a beam in pure bending. Taking advantage of symmetry, a quarter of the beam is modeled using plane stress elements, see figure 1(b). The beam was analyzed for a range of different thicknesses while keeping the length constant. This way the element aspect ratio (ratio of the largest dimension to the smallest) can be varied. Again, the condition of A increases with the aspect ratio. We refer to this as geometric ill-conditioning. Three different levels of mesh refinement were used to study the effect of problem size on the algorithms ; 4×16, 8×32 and 16×64.

		Lanczos with Partial Reorthogonalization						Conjugate Gradients		
		E×E LU Split		E×E Choleski		Diagonal		E×E LU Split	E×E Choleski	Diagonal
Penalty Param.	κ	# Iter.	Reorth. Cost*	# Iter.	Reorth. Cost*	# Iter.	Reorth. Cost*	# Iter.	# Iter.	# Iter.
10^4	10^6	236	19445	228	17245	393	95424	473	424	792
10^3	10^5	193	11134	184	8900	348	60869	277	261	517
10^2	10^4	117	440	110	768	247	21393	117	110	252
10^1	10^3	40	0	39	0	98	708	40	39	98
1	10^2	26	0	25	0	60	0	26	25	60

* Unit is one dot product and one SAXPY (vector plus a scalar times a vector).

Table 3. Result of tests using 20×20 mesh. $n = 722$.

To illustrate the advantages of semi-orthogonality we evaluate the solution for the 4×16 beam problem using the CG method, Lanczos method with full reorthogonalization, and Lanczos with Simon's scheme for maintaining semi-orthogonality. In figure 2 we plot the residual norm against the iteration number for each of the methods. The results for the two implementations of the Lanczos method are indistinguishable. The residual norm in the CG method starts off the same as that in the Lanczos method, but it deviates quickly and takes four times as many steps to converge. The difference in the curves for the CG and Lanczos is due to loss of orthogonality in the CG method.

We use the 20×20 Stokes flow problem to make a direct comparison between three different preconditioners; diagonal scaling, the E×E Choleski defined in (30), and the E×E LU split defined in (31). We solve this problem for a range of different penalty parameters using both Lanczos and CG methods. A summary of the results is given in Table 3. Sample plots of the residual norm against the iteration number are illustrated in Figures 3 and 4. The number of iterations required to obtain the solution using E×E LU split is marginally more than that using E×E Choleski. On average the cost of reorthogonalization for E×E Choleski was slightly less. On the other hand, E×E Choleski requires additional storage to keep the preconditioning matrix. It is interesting to note that the number of reorthogonalizations increases with the penalty parameter (condition number). This indicates that Simon's scheme performs reorthogonalizations when ever it is needed. The number of iterations given in Table 3 are plotted against the penalty parameter; see Figure 5. One can observe from this plot that maintaining orthogonality can results in reductions of factors of two in the number of iterations for this example.

			Lanczos with Partial Reorthogonalization				Conjugate Gradients	
			E×E LU Split		Diagonal		E×E LU Split	Diagonal
Problem	Aspect Ratio	κ	No. of Iter.	Reorth. Cost*	No. of Iter.	Reorth. Cost*	No. of Iter.	No. of Iter.
	1	10^4	146	511	378	17352	182	506
16×64	2	2×10^5	199	1208	542	106880	344	1027
$n = 2142$	4	3×10^6	329	3261	896	296617	850	2336
	8	5×10^7	586	47667	1132+	811600+	2714	6000+
	40	3×10^{10}	886	217883	1648+	1317042+	6000+	6000+
	1	2×10^2	41	146	89	2382	41	96
4×16	2	3×10^3	56	196	111	5625	71	236
$n = 151$	4	5×10^4	87	624	141	13951	191	507
	8	7×10^5	114	2312	151	17482	605	1532
	40	5×10^8	150	6461	151	17784	2216	5614

* Unit is one dot product and one SAXPY (vector plus a scalar times a vector).

Table 4. Result of tests using the beam in pure bending. + indicates that CPU time exceeded.

We obtain a similar set of results for the beam example; see Table 4. Both diagonal and E×E LU split are used to precondition the problem. The solution is evaluated for three different levels of discretization

using Lanczos and CG methods. Sample plots for the 8×32 mesh are illustrated in Figure 6 for thickness $t = 1.0$ and in Figure 7 for $t = 0.1$. When $t = 1.0$, CG method required three times as many steps to converge as the Lanczos method. More crucial is the fact that CG failed to converge after 6000 iterations when $t \leq 0.5$, even for E×E preconditioners. On the other hand Lanczos delivered the solution in less than 300 iterations using E×E preconditioners.

The 16×64 beam problem was also analyzed using CG method with S×S preconditioner using $t = 1.0$ and $t = 0.05$ corresponding to element aspect ratios 1 and 80, respectively. The mesh is partitioned into 4, 8, 16, and 32 substructures. All the partition lines were through the thickness of the beam. These partitions were chosen with equal number of elements in each substructure to illustrate the performance of the S×S solution algorithm. The computations were carried out on a hypercube concurrent computer having a total of 32 processors. Each substructure was assigned to a different processor. Thus, when the number of partitions is 8, only 8 of the processors in the hypercube is utilized. The results for these analysis are given in Table 5.

Beam Thickness	No. of Substructures	Solution Time (Sec.)	Elimination Time (Sec.)	PCG Time (Sec.)	No. of Iterations
1.0	4	91	82	9	58
	8	53	37	16	95
	16	40	14	26	148
	32	43	3	40	228
0.05	4	110	82	28	164
	8	80	37	43	253
	16	68	14	54	308
	32	76	3	73	410

Table 5. Result of CG with S×S Preconditioner for the beam in pure bending.

The elimination of the interior unknows in each substructure was carried out using a direct method (profile solver). As the mesh is partitioned into more substructures, the number of unknowns in the interior decreases. For the partitioning scheme used for this problem, the bandwidth of the substructure stiffness matrix does not change with the number partitions. As a result, the parallel time for eliminating all the interior degrees of freedom becomes inversely proportional to the number of substructures. Moreover, the number of boundary nodes in each substructure remains constant. A direct consequence of this is that the total number of interface node and thus the number of equations in the reduced system solved by PCG become directly proportional to the number of substructure. Then the parallel time for each PCG iteration remains constant. For the S×S preconditioners, the number of PCG iterations depends on the number of equations and for this choice of partitions, the total parallel PCG time is proportional to the square root of the number of processors.

This example clear indicates that there is an optimum number of processors for a given problems that minimizes the total solution time (direct + PCG). For the beam problem this minimum occurs for 16 processors. This optimum processor number is expected to increase with the problem size.

8. Conclusions

The conjugate gradient method may be derived directly from the Lanczos algorithm by performing an implicit triangular factorization of the resulting reduced tridiagonal matrix. So long as this factorization is numerically stable the conjugate gradient will also be stable. For symmetric positive definite systems this factorization is stable. In the case of indefinite equations the success of the conjugate gradient method can not be guaranteed. However, the Lanczos process will always converge for all symmetric systems.

Partial reorthogonalization improves the robustness of the Lanczos algorithm so that it always converges. However, for ill conditioned system the cost of partial reorthogonalization can be substantial. For well conditioned problems where there is no tendency towards loss of orthogonality among the Lanczos vectors, partial reorthogonalization will marginally increase the cost of the Lanczos algorithm. Therefore, partial reorthogonalization should always be used in conjunction with good preconditioners. It will pick up the slack for preconditioners.

The $S \times S$ preconditioner is introduced as an extension of the $E \times E$ preconditioner. It is a hybrid method combining direct elimination of degrees of freedom interior to substructures with iterative solution for the unknowns on the boundary nodes of substructures. The $S \times S$ preconditioners are always more effective than the $E \times E$ techniques.

9. References

[1] C. Lanczos, "An Iteration Method for the Solution of the Eigenvalue Problems of Linear Differential and Integral Operators," *J. Res. Nat. Bur. Standards*, **45** (1950), pp. 255-282.

[2] M. R. Hestenes and E. Stiefel, "Method of Conjugate Gradient for Solving Linear Systems," *J. Res. Nat. Bur. Standards*, **45** (1952), pp. 409-436.

[3] C. Lanczos, "Solution of Systems of Linear Equations by Minimized Iteration," *J. Res. Nat. Bur. Standards*, **49** (1952), pp. 33-53.

[4] R. K. Lively, "The Analysis of Large Structural Systems," *Computer J.*, **3** (1960), pp. 34-39.

[5] O. C. Zienkiewicz, *The Finite Element Method*, Third Edition , Mc-Graw Hill, Inc., London, 1977.

[6] A. George and J. W. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice Hall, Englewood Cliffs, NJ, 1981.

[7] J. K. Reid, "On the Method of Conjugate Gradients for the Solution of Large Sparse Systems of Linear Equations" in *Large Sparse Sets of Linear Equations*, Academic Press, New York, 1971, pp. 231-254.

[8] R. L. Fox and E. L. Stanton, "Developments in Structured Analysis by Direct Energy Minimization," *AIAA Journal*, **6** (1968), pp. 1036-1042.

[9] I. Fried, "More on Gradient Iterative Methods in Finite-Element Analysis," *AIAA Journal*, **7** (1969), pp. 565-567.

[10] T. J. R. Hughes, I. Levit and J. Winget, "Implicit, Unconditionally Stable Algorithms for Heat Conduction Analysis," *ASCE, Journal of the Engineering Mechanics Division*, **109** (1983), pp. 576-585.

[11] M. Ortiz, P. M. Pinsky and R. L. Taylor, "Unconditionally Stable Element-by-Element algorithm for Dynamic Problems," *Computer Methods in Applied Mechanics and Engineering*, **36** (1983), pp. 223-239.

[12] T. J. R. Hughes, I. Levit and J. Winget, "An Element-by-Element Solution Algorithms for Problems of Structural and Solid Mechanics," *Computer Methods in Applied Mechanics and Engineering*, **36** (1983), pp. 241-254.

[13] B. Nour-Omid and B. N. Parlett, "Element Preconditioning Using Splitting Techniques," *SIAM J. Sci. Stat. Comput.*, **6** (1985), pp. 761-770.

[14] J. M. Winget and T. J. R. Hughes, "Solution Algorithms for Nonlinear Transient Heat Conduction Analysis Employing Element-by-Element Iterative Strategies, *Computer Methods in Applied Mechanics and Engineering*, **52** (1985), pp. 711-815.

[15] D. S. Kershaw, "The Incomplete Choleski-Conjugate Gradient Method for Solution of System of Linear Equations," *Journal of Computational Physics*, **26** (1978), pp. 43-65.

[16] H. D. Simon, "The Lanczos Algorithm with Partial Reorthogonalization," *Mathematics of Computation*, **42** (1984), pp. 115-142.

[17] B. N. Parlett, "A New Look at the Lanczos Algorithm for Solving Symmetric Systems of Linear Equations," *Linear Algebraic Applications*, Vol. 29, pp. 323-346, 1980.

[18] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 1983.

[19] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1980.

[20] B. Nour-Omid, "A Preconditioned Conjugate Gradient Method for Solution of Finite Element Equations," *Innovative Methods for Nonlinear Problems, ASME*, New Orleans, Dec. 1984.

[21] E. L. Wachspress, "Extended Application of Alternating Direction Implicit Iteration Model Problem Theory," *J. Soc. Industr. Appl. Math.*, **11** (1963), pp. 994-1016.

[22] A. R. Gourlay, "Splitting Methods for Time Dependent Partial Differential Equations", in *The State of the Art in Numerical Analysis*, ed. by D. Jacobs, Academic Press, 1977.

[23] M. R. Li, B. Nour-Omid, and B. N. Parlett, "A Fast Solver Free of Fill-In for Finite Element Problems," *SIAM J. Numer. Anal.*, Vol. 19, No. 6, Dec. 1982, pp. 1233-1242.

[24] M. Ortiz, and B. Nour-Omid, "Unconditionally Stable Concurrent Procedures for Transient Finite Element Analysis," *Computer Methods in Applied Mechanics and Engineering*, Vol. 58, pp. 151-174, 1986.

[25] M. Ortiz, B. Nour-Omid, and E. D. Sotolino, "Accuracy of a Class of Concurrent Algorithms for Transient Finite Element Analysis," *International Journal for Numerical Methods in Engineering*, Vol. 26, pp. 379-391, 1988.

[26] B. Nour-Omid, and M. Ortiz, "A Family of Concurrent Algorithm for Transient Finite Element Analysis," *Proceedings of the ASCE Structures Congress on Parallel Processing and Computational Strategies in Structural Engineering*, May 1989.

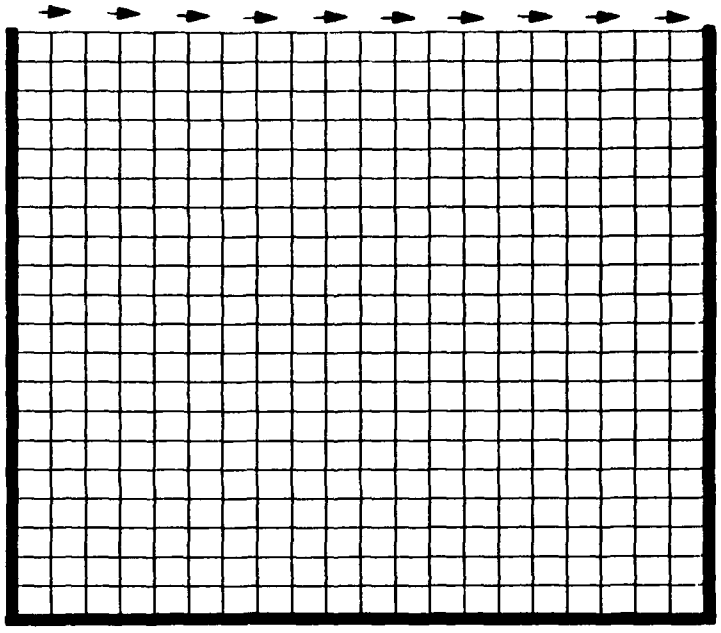
[27] B. Nour-Omid, A. Raefsky and G. Lyenga, "Solving Finite Element Equations on Concurrent Computers," *Parallel Computations and their Impact on Mechanics*, A. Noor (ed.) ASME, Boston, Dec. 13-18 1987.

[28] B. Nour-Omid, and K. C. Park, "Solving Structural Mechanics Problems on the Caltech Hypercube Machine," *Computer Methods in Applied Mechanics and Engineering*, Vol. 61, pp. 161-176, 1987.

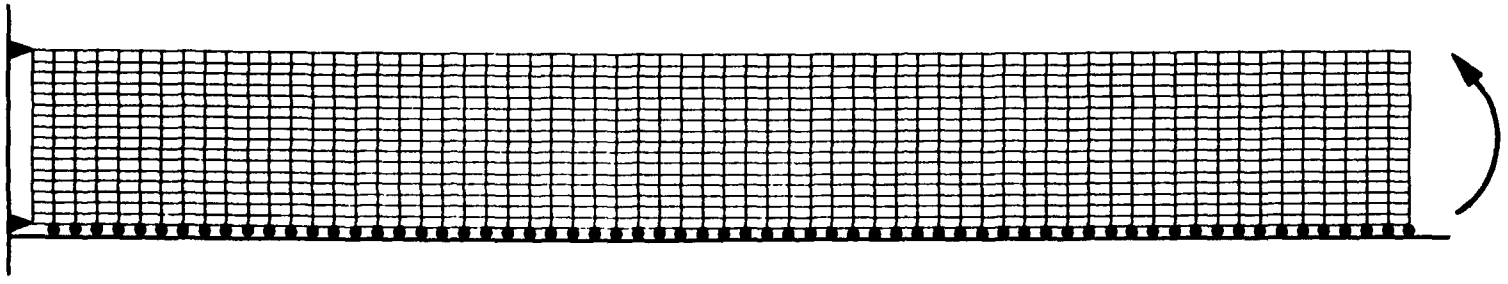
Figure 1.

(a) 20×20 mesh for the incompressible fluid flow in a cavity.

(b) 16×64 mesh for the beam in pure bending.



(a)



(b)

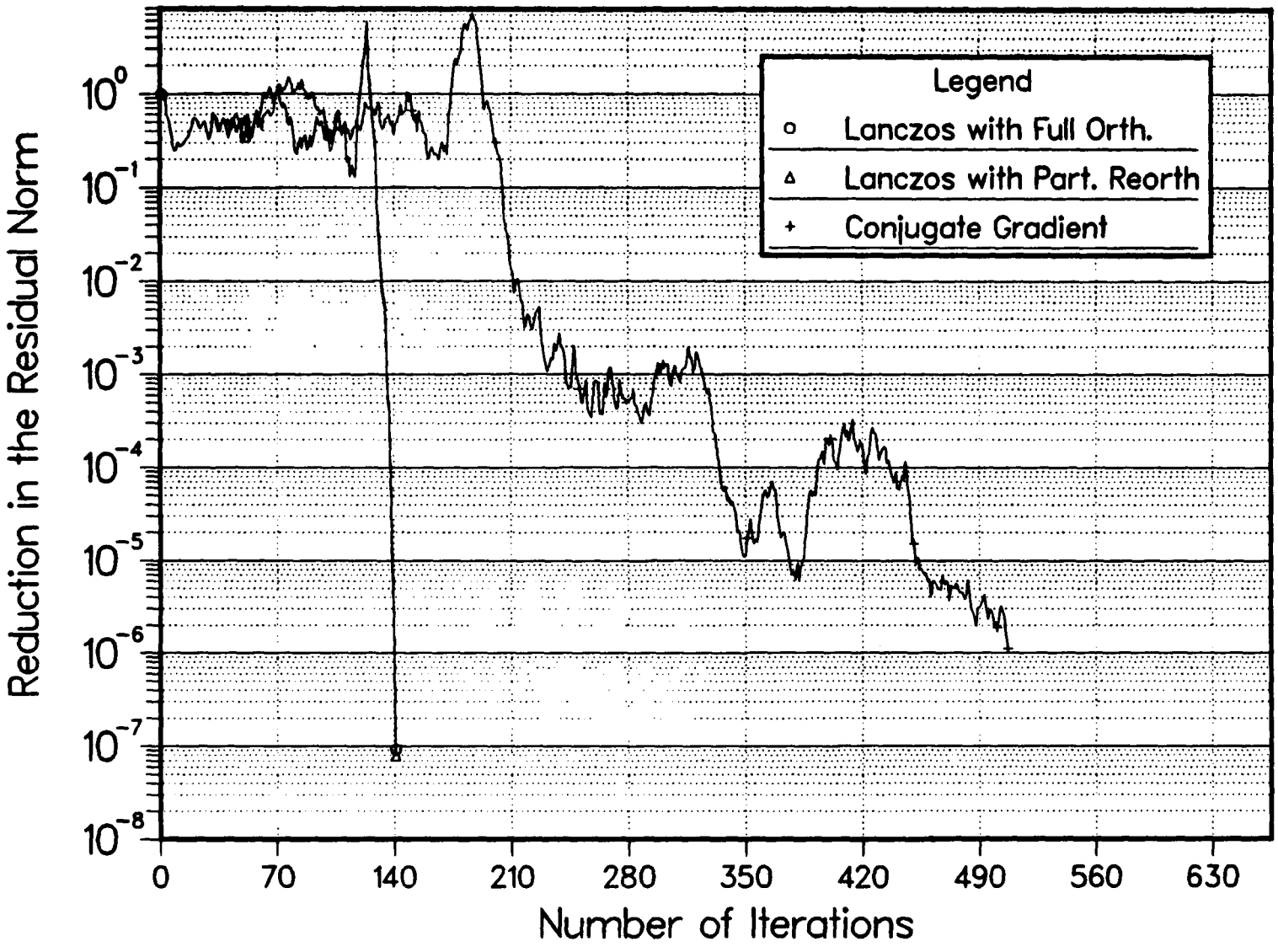


Figure 2. Effect of reorthogonalization on the number of iterations; 4×16 beam with $t = 1.0$ and diagonal preconditioning.

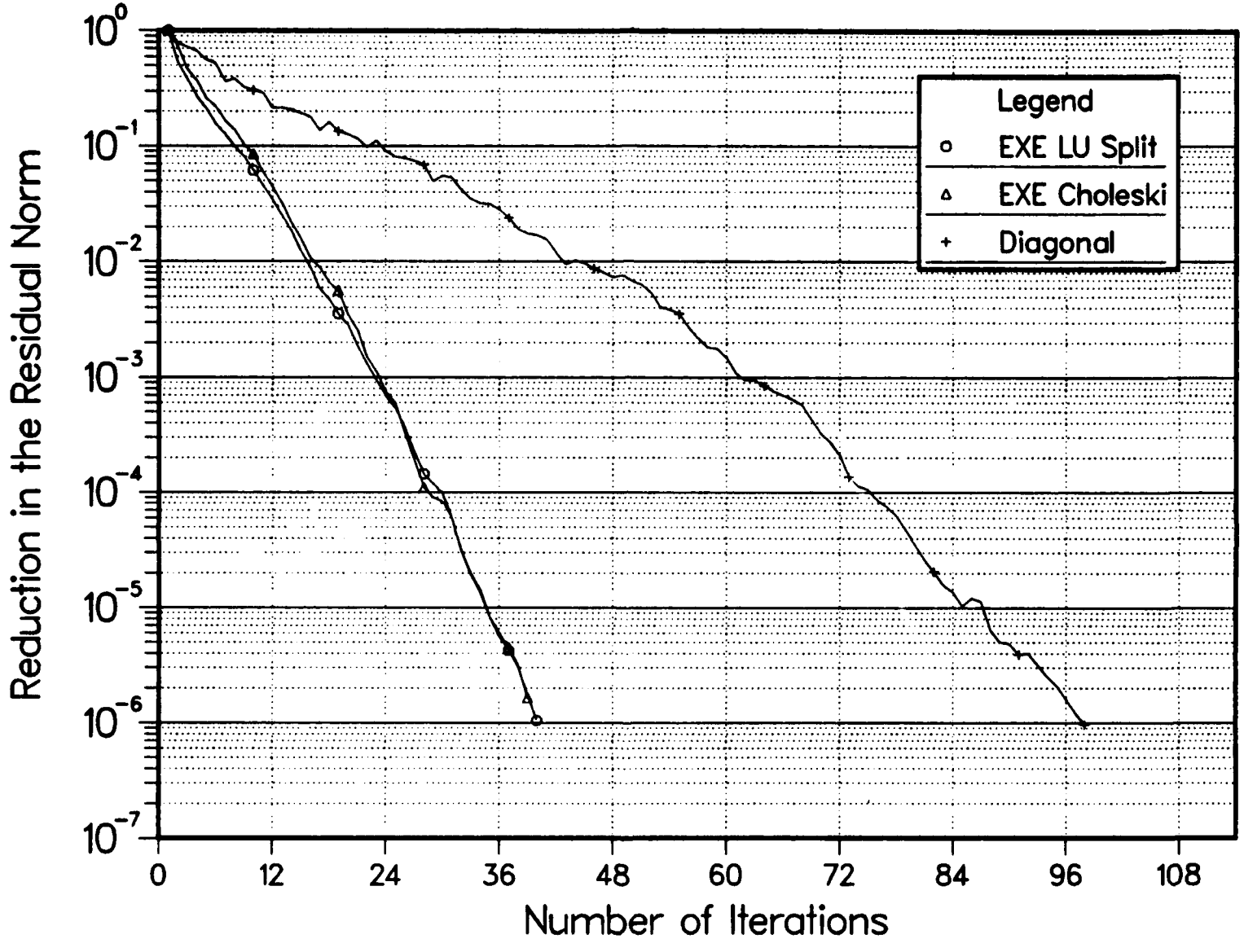


Figure 3. Effect of preconditioning on the number of Lanczos steps for 20×20 mesh with penalty parameter of 10.

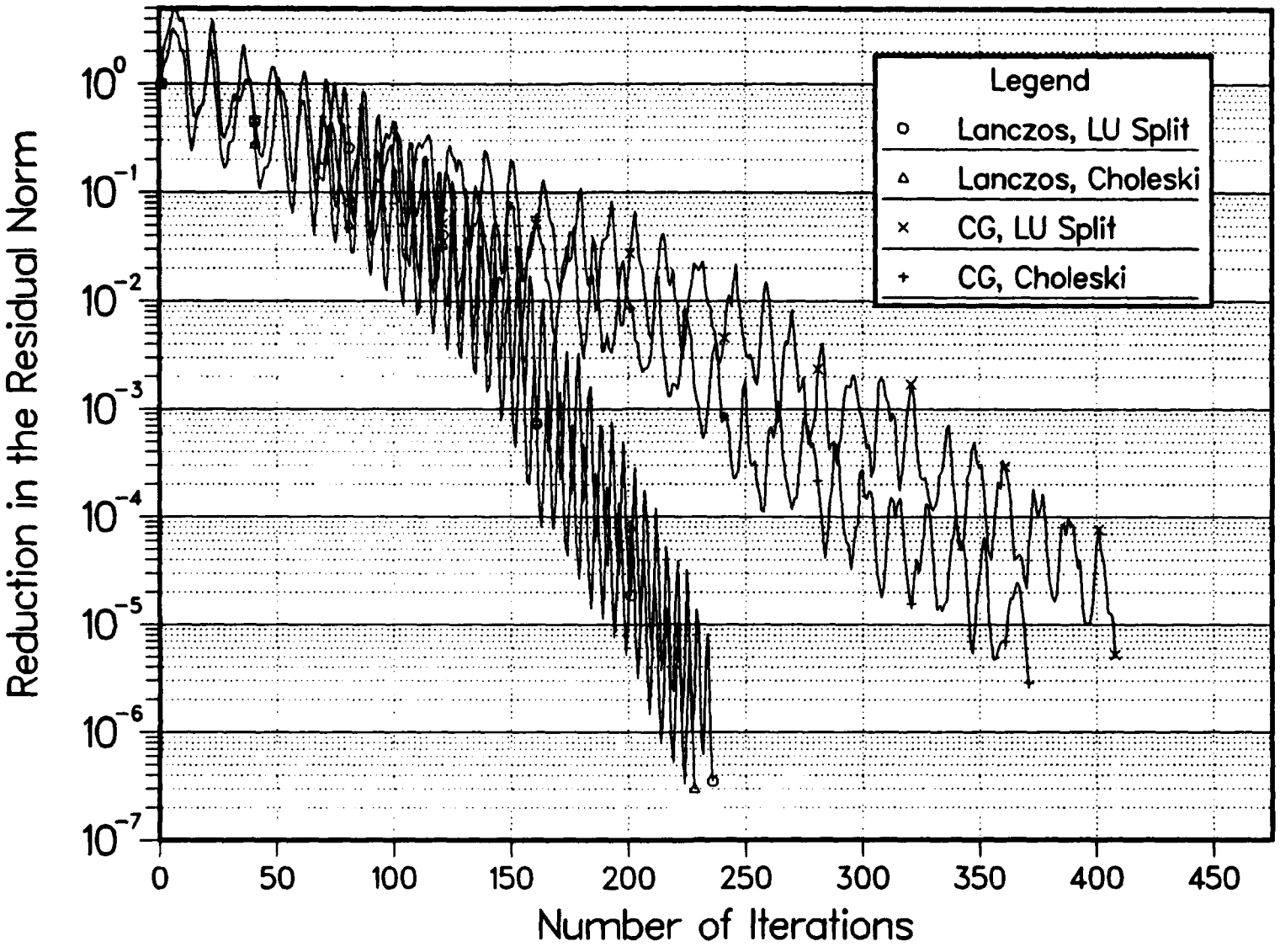


Figure 4. Comparison of different E×E preconditioners using the 20×20 mesh with penalty parameter of 10^4.

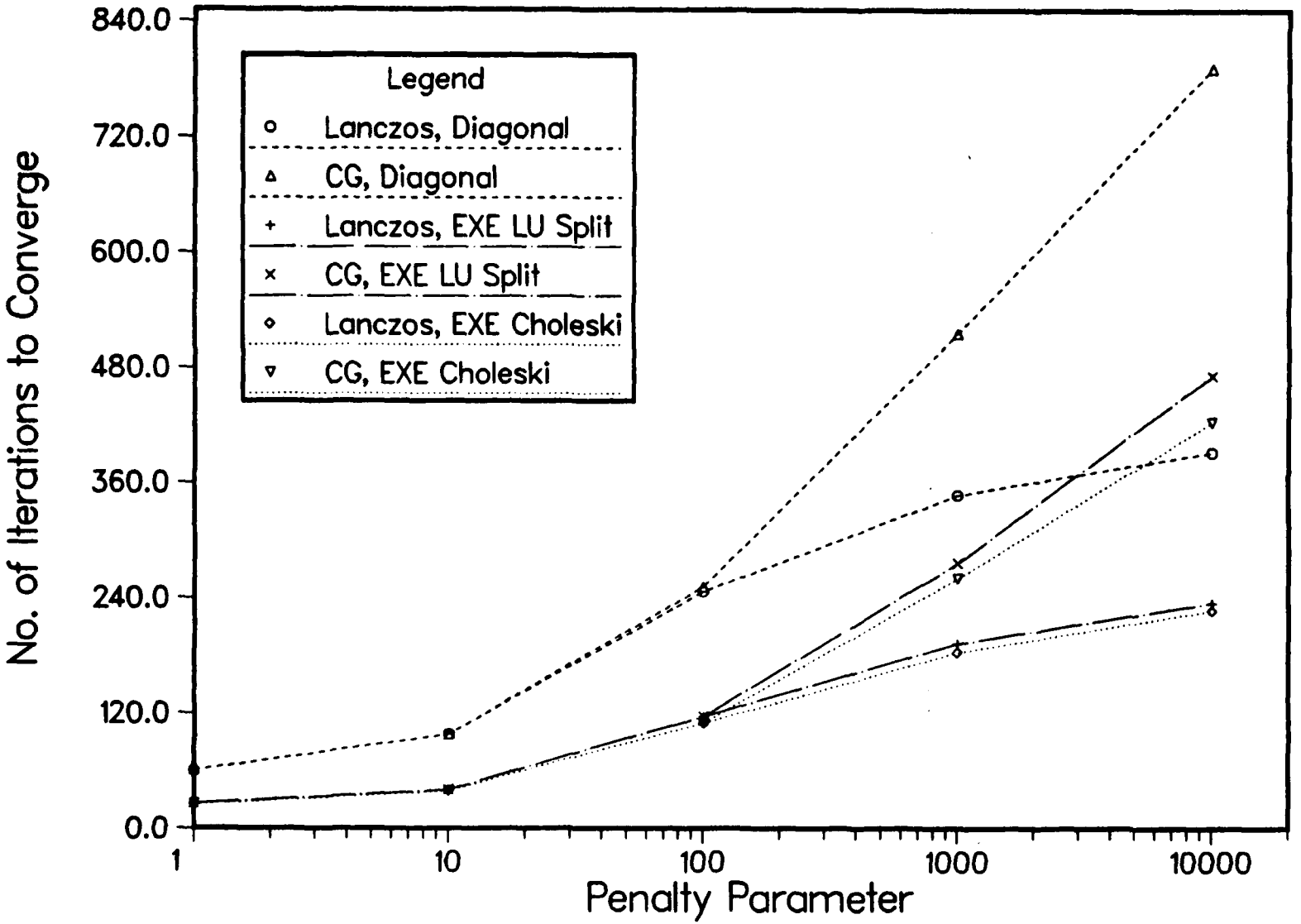


Figure 5. Effect of illconditioning due to element properties on Lanczos and CG methods (20×20 mesh).

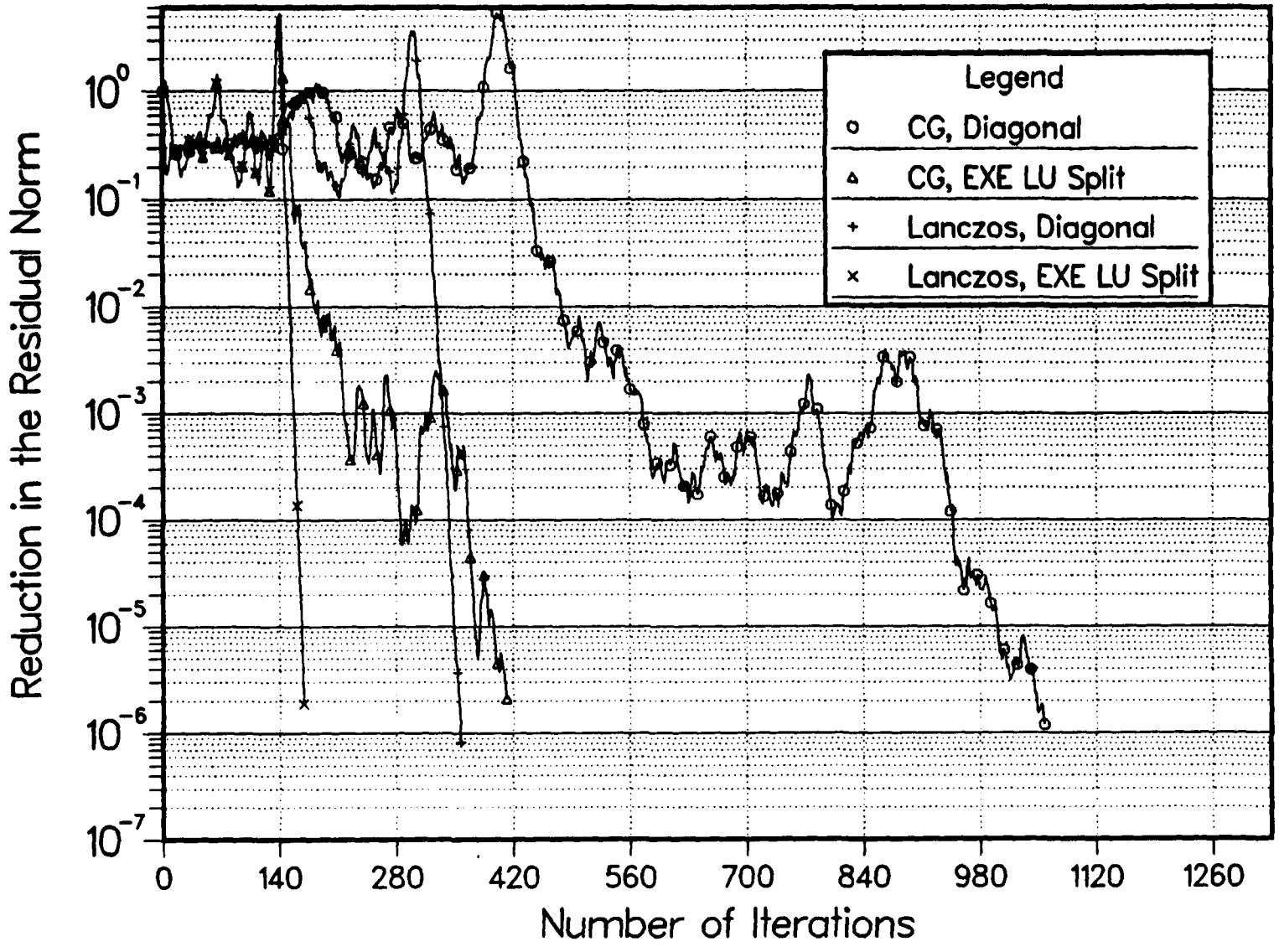


Figure 6. 8×32 beam problem with $t = 1.0$; element aspect ratio = 4.

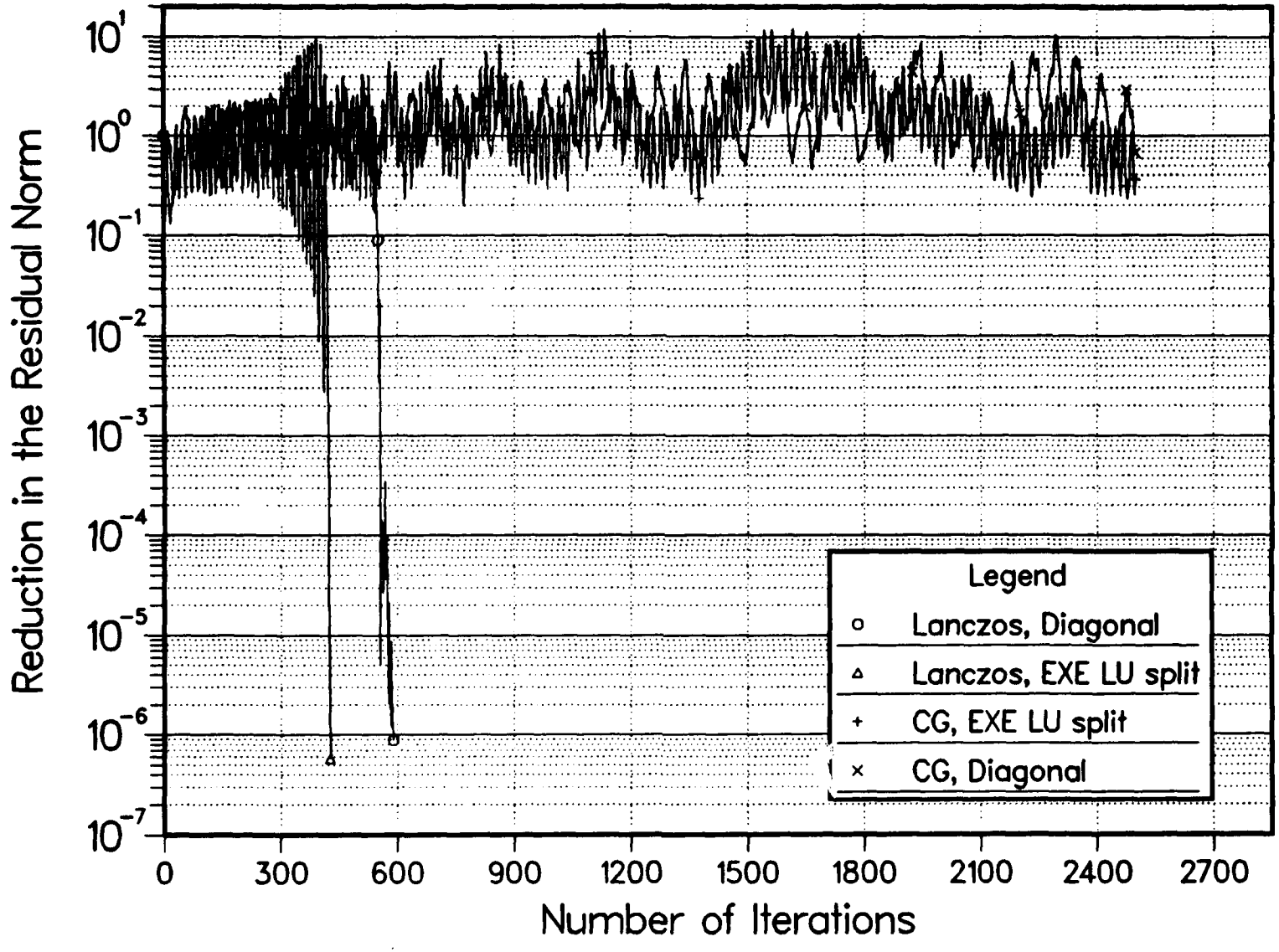


Figure 7. 8×32 beam problem with $t = 0.1$; element aspect ratio = 40.

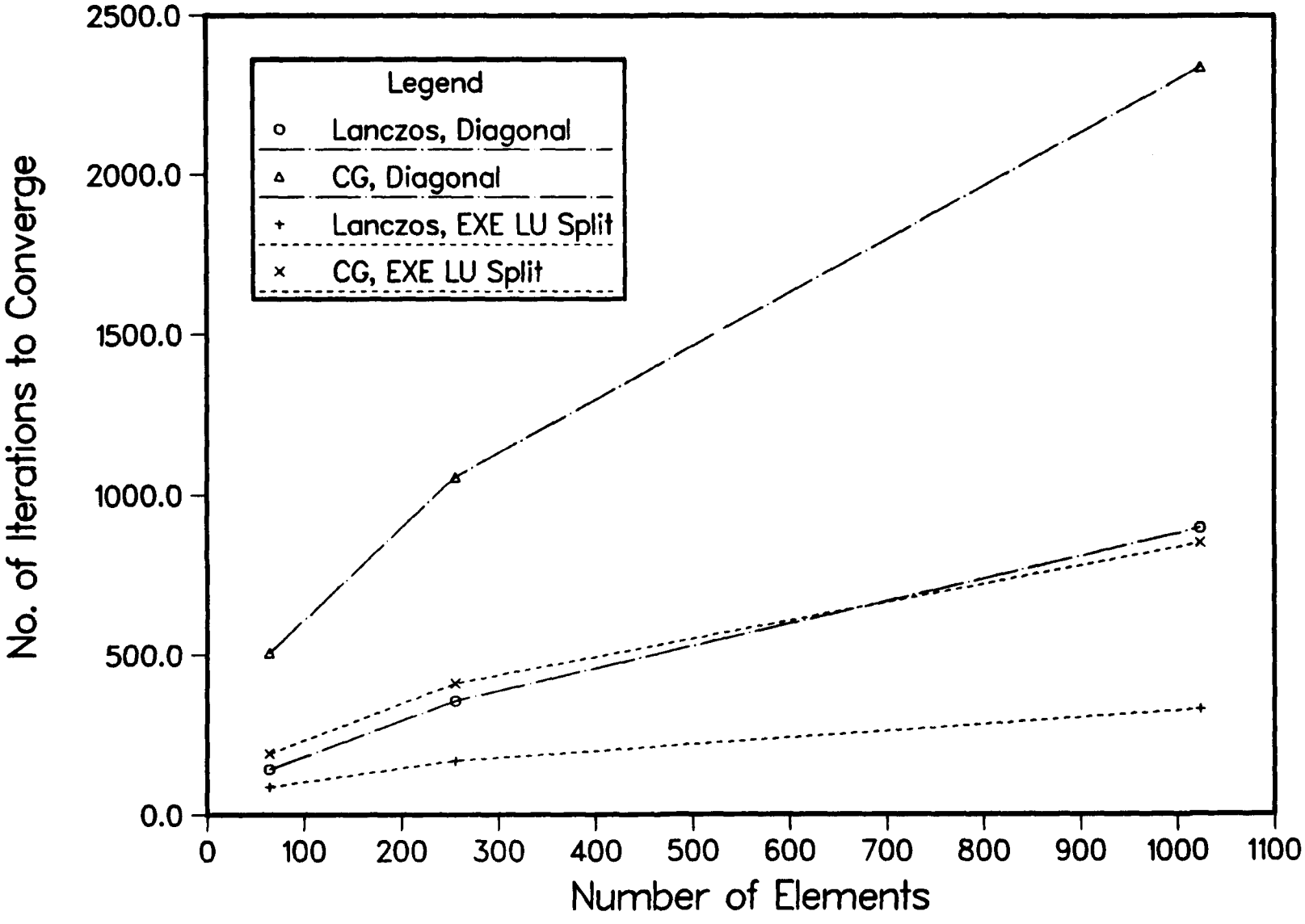


Figure 8. Effect of illconditioning due to mesh refinement on Lanczos and CG methods for the beam problem.

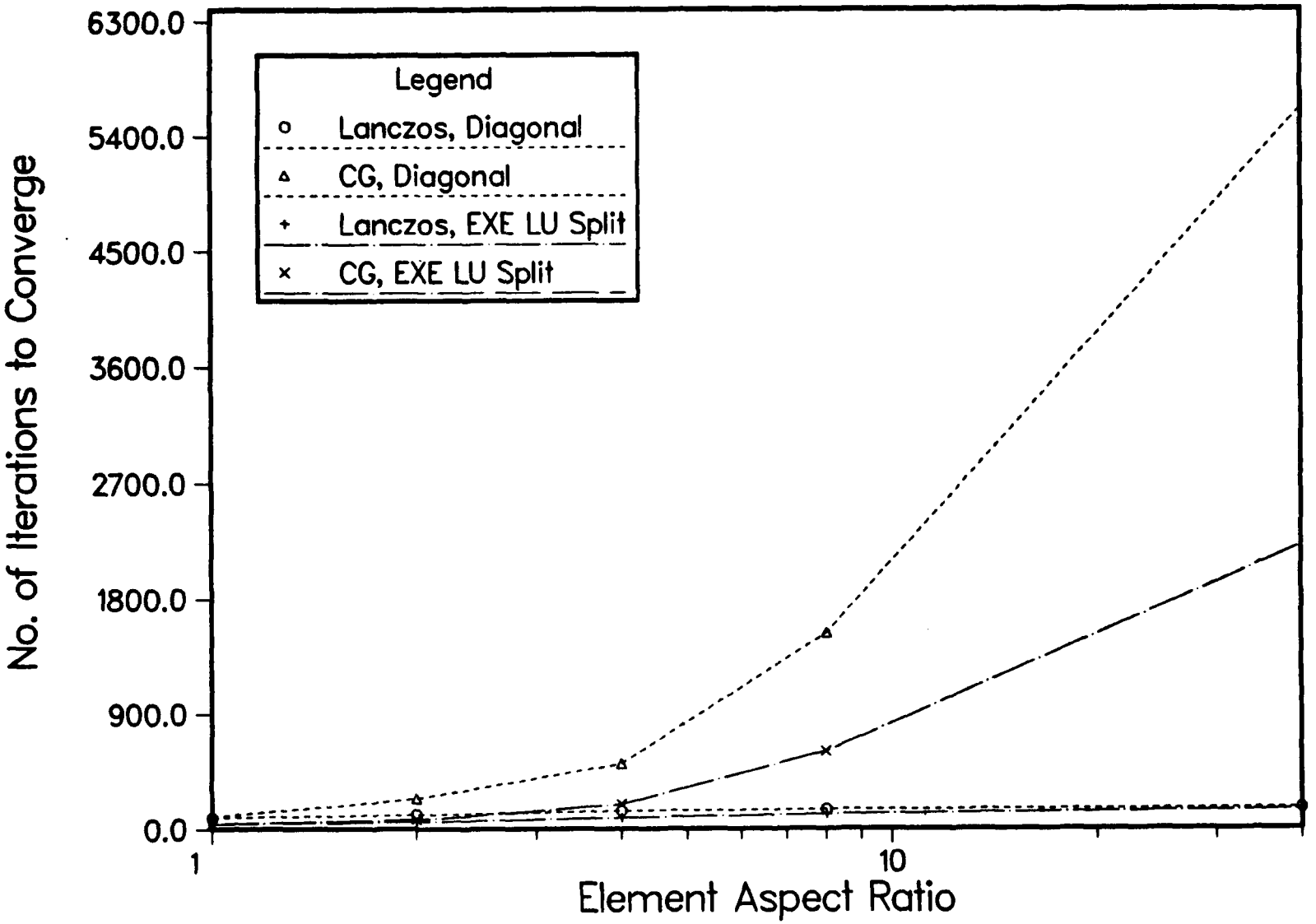


Figure 9. Effect of illconditioning due to element aspect ratio on Lanczos and CG methods; 4×16 beam.

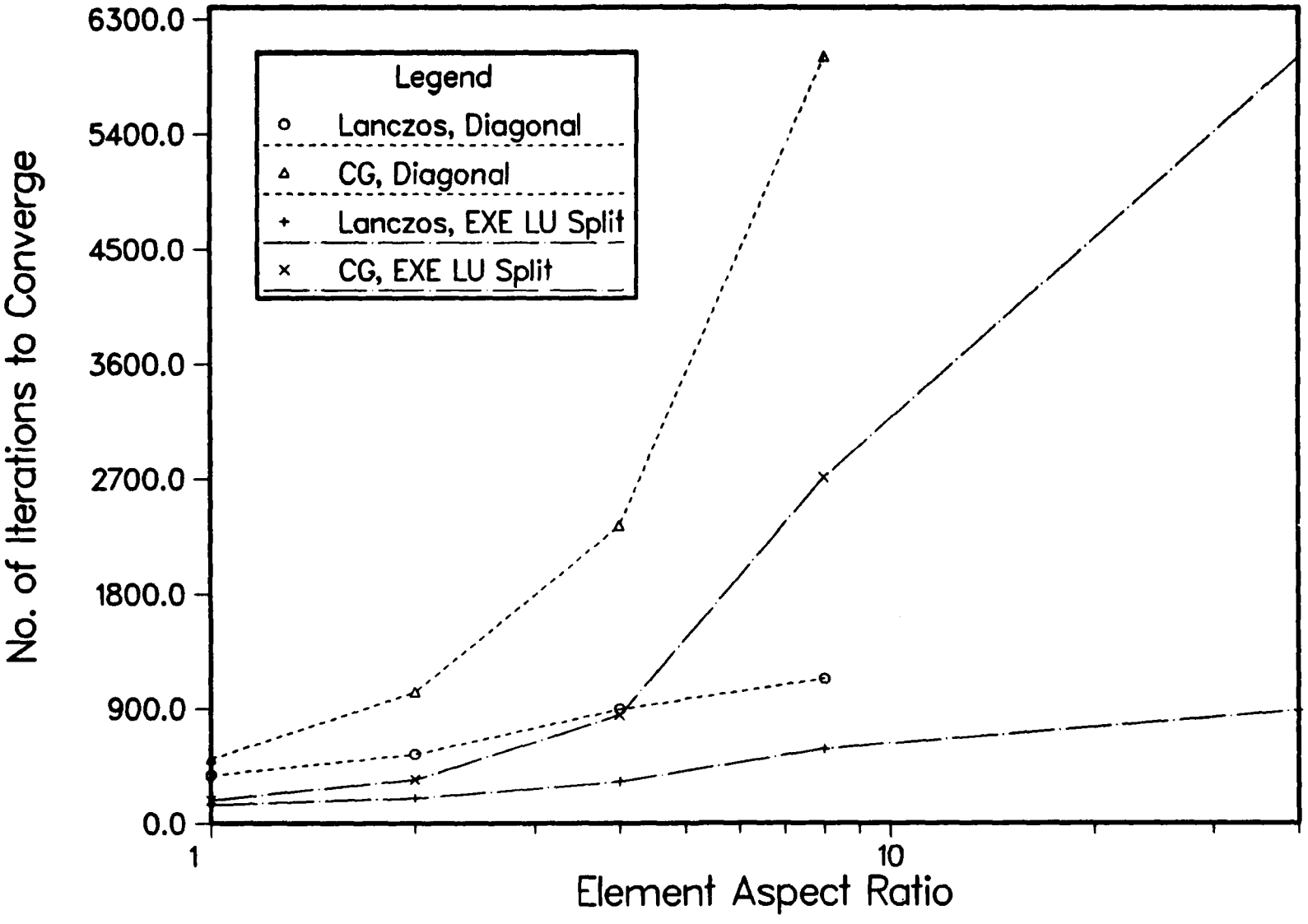


Figure 10. Effect of illconditioning due to element aspect ratio on Lanczos and CG methods; 16×64 beam.

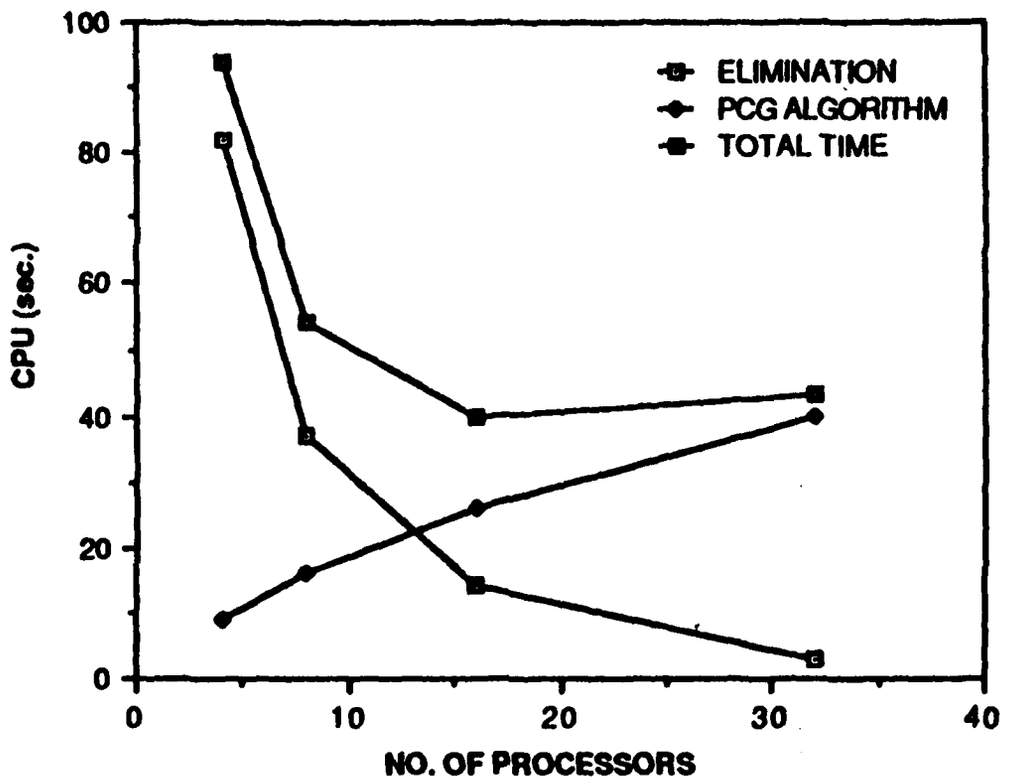


Figure 11. S×S PCG on a hypercube concurrent computer; 16×64 beam with $t = 1.0$.

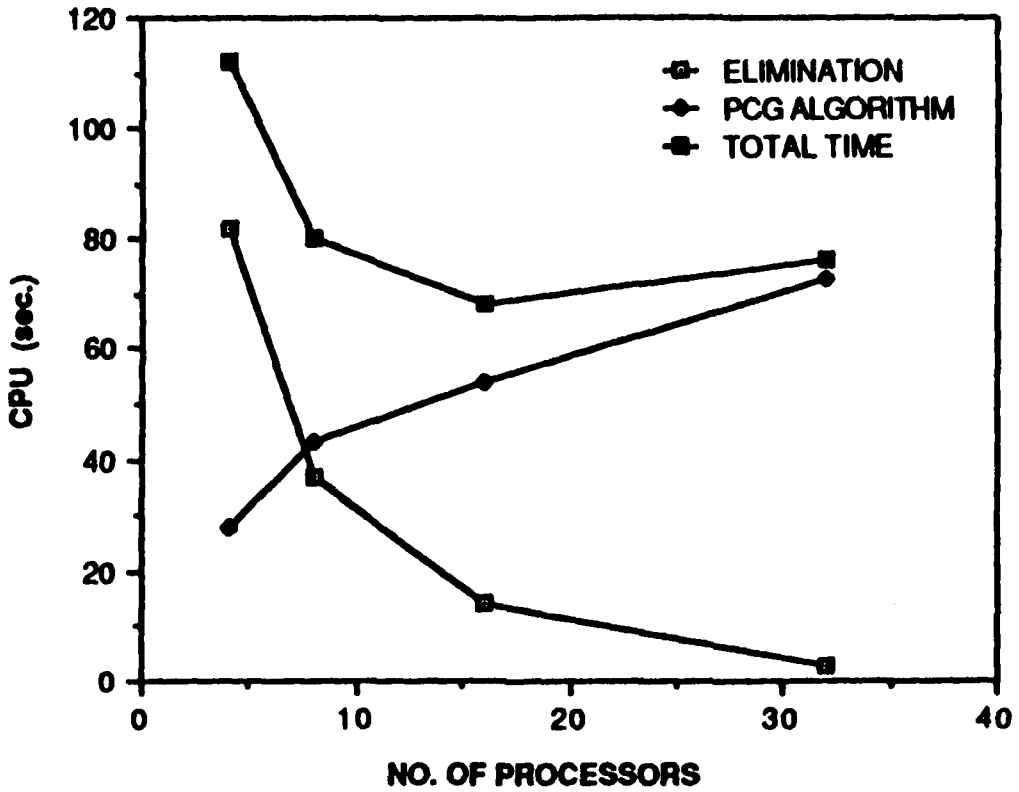


Figure 12. S×S PCG on a hypercube concurrent computer; 16×64 beam with $t = 0.05$.

DISTRIBUTION LIST

ADINA ENGRG, INC / WALCZAK, WATERTOWN, MA
AFOSR / NA (WU), WASHINGTON, DC
APPLIED RSCH ASSOC, INC / HIGGINS, ALBUQUERQUE, NM
APTEK / SCHWER, SAN JOSE, CA
ARMSTRONG AERO MED RSCH LAB / OVENSHIRE, WRIGHT PATTERSON AFB, OH
ARMY CORPS OF ENGRS / HQ, DAEN-ECE-D, WASHINGTON, DC
ARMY EWES / WES (NORMAN), VICKSBURG, MS; WES (PETERS), VICKSBURG, MS; WESIM-C (N. RADHAKRISHNAN), VICKSBURG, MS
CATHOLIC UNIV / CE DEPT (KIM) WASHINGTON, DC
CENTRIC ENGINEERING SYSTEMS INC / TAYLOR, PALO ALTO, CA
DOT / TRANSP SYS CEN (TONG), CAMBRIDGE, MA
DTIC / ALEXANDRIA, VA
DTRCEN / (CODE 1720), BETHESDA, MD
GEN MOTORS RSCH LABS / (KHALIL), WARREN, MI
GEORGIA INST OF TECH / MECH ENGRG (FULTON), ATLANTA, GA
HKS INC / LOOP NAGTEGAAL PROVIDENCE, RI
HQ AFESC / RDC (DR. M. KATONA), TYNDALL AFB, FL
LAWRENCE LIVERMORE NATIONAL LAB / WHIRLEY, LIVERMORE, CA
LOCKHEED / RSCH LAB (M. JACOBY), PALO ALTO, CA; RSCH LAB (P UNDERWOOD), PALO ALTO, CA
MARC ANALYSIS RSCH CORP / HSU, PALO ALTO, CA
MEDWADOWSKI, S. J. / CONSULT STRUCT ENGR, SAN FRANCISCO, CA
NAVFACENGCOM / CODE 04B2 (J. CECILIO), ALEXANDRIA, VA; CODE 04BE (WU), ALEXANDRIA, VA
NORTHWESTERN UNIVERSITY / BAZANT, EVANSTON, IL; CE DEPT (BELYTSCHKO), EVANSTON, IL
NRL / CODE 4430, WASHINGTON, DC
NSF / STRUC & BLDG SYSTEMS (KP CHONG), WASHINGTON, DC
NUSC DET / CODE 44 (CARLSEN), NEW LONDON, CT
OCNR / CODE 10P4 (KOSTOFF), ARLINGTON, VA; CODE 1121 (EA SILVA), ARLINGTON, VA; CODE 1132SM, ARLINGTON, VA
OHIO STATE UNIV / CE DEPT (SIERAKOWSKI), COLUMBUS, OH
ONR / CODE 1131S, ARLINGTON, VA; CODE 1132SM, ARLINGTON, VA; CODE 6000, ARLINGTON, VA
OREGON STATE UNIV / CE DEPT (HUDSPETH), CORVALLIS, OR; CE DEPT (LEONARD), STORRS, CT; CE DEPT (YIM), CORVALLIS, OR; DEPT OF MECH ENGRG (SMITH), CORVALLIS, OR
PORTLAND STATE UNIV / ENGRG DEPT (MIGLIORI), PORTLAND, OR
SCOPUS TECHNOLOGY INC / (B NOUR-OMID), EMERYVILLE, CA; (S. NOUR-OMID), EMERYVILLE, CA
SRI INTL / ENGRG MECH DEPT (GRANT), MENLO PARK, CA; ENGRG MECH DEPT (SIMONS), MENLO PARK, CA
STANFORD UNIV / APP MECH DIV (HUGHES), STANFORD, CA; CE DEPT (PENSKY), STANFORD, CA; DIV OF APP MECH (SIMO), STANFORD, CA
TRW INC / CRAWFORD, REDONDO BEACH, CA
TUFTS UNIV / SANAYEI, MEDFORD, MA

UNIV OF CALIFORNIA / CE DEPT (HERRMANN), DAVIS, CA; CE DEPT (KUTTER), DAVIS, CA; CE DEPT (RAMEY), DAVIS, CA; CE DEPT (ROMSTAD), DAVIS, CA; CE DEPT (WILSON), BERKELEY, CA; CTR FOR GEOTECH MODEL (IDRISS), DAVIS, CA; FOURNEY, LOS ANGELES, CA; MECH ENGRG DEPT (BAYO), SANTA BARBARA, CA; (BRUCH), SANTA BARBARA, CA; (LECKIE), SANTA BARBARA, CA; (MCMEEKING), SANTA BARBARA, CA; (MITCHELL), SANTA BARBARA, CA; (TULIN), SANTA BARBARA, CA; SELMA, LOS ANGELES, CA

UNIV OF COLORADO / CE DEPT (HON-YIM KO), BOULDER, CO; MECH ENGRG DEPT (FELLIPA), BOULDER, CO; MECH ENGRG DEPT (PARK), BOULDER, CO

UNIV OF ILLINOIS / CE LAB (ABRAMS), URBANA, IL; CE LAB (PECKNOLD), URBANA, IL

UNIV OF N CAROLINA / CE DEPT (GUPTA), RALEIGH, NC; CE DEPT (TUNG), RALEIGH, NC

UNIV OF TEXAS / CE DEPT (STOKOE), AUSTIN, TX

UNIV OF WYOMING / CIVIL ENGRG DEPT, LARAMIE, WY

WEBSTER, R / BRIGHAM CITY, UT

WEIDLINGER ASSOC / F.S. WONG, LOS ALTOS, CA

DISTRIBUTION QUESTIONNAIRE

The Naval Civil Engineering Laboratory is revising its primary distribution lists.

SUBJECT CATEGORIES

1 **SHORE FACILITIES**
1A Construction methods and materials (including corrosion control, coatings)
1B Waterfront structures (maintenance/deterioration control)
1C Utilities (including power conditioning)
1D Explosives safety
1E Aviation Engineering Test Facilities
1F Fire prevention and control
1G Antenna technology
1H Structural analysis and design (including numerical and computer techniques)
1J Protective construction (including hardened shelters, shock and vibration studies)
1K Soil/rock mechanics
1L Airfields and pavements
1M Physical security
2 **ADVANCED BASE AND AMPHIBIOUS FACILITIES**
2A Base facilities (including shelters, power generation, water supplies)
2B Expedient roads/airfields/bridges
2C Over-the-beach operations (including breakwaters, wave forces)
2D POL storage, transfer, and distribution
2E Polar engineering
3 **ENERGY/POWER GENERATION**
3A Thermal conservation (thermal engineering of buildings, HVAC systems, energy loss measurement, power generation)
3B Controls and electrical conservation (electrical systems, energy monitoring and control systems)
3C Fuel flexibility (liquid fuels, coal utilization, energy from solid waste)
3D Alternate energy source (geothermal power, photovoltaic power systems, solar systems, wind systems, energy storage systems)
3E Site data and systems integration (energy resource data, integrating energy systems)
3F EMCS design
4 **ENVIRONMENTAL PROTECTION**
4A Solid waste management
4B Hazardous/toxic materials management
4C Waterwaste management and sanitary engineering
4D Oil pollution removal and recovery
4E Air pollution
4F Noise abatement
5 **OCEAN ENGINEERING**
5A Seafloor soils and foundations
5B Seafloor construction systems and operations (including diver and manipulator tools)
5C Undersea structures and materials
5D Anchors and moorings
5E Undersea power systems, electromechanical cables, and connectors
5F Pressure vessel facilities
5G Physical environment (including site surveying)
5H Ocean-based concrete structures
5J Hyperbaric chambers
5K Undersea cable dynamics

ARMY FEAP
BDG Shore Facilities
NRG Energy
ENV Environmental/Natural Responses
MGT Management
PRR Pavements/Railroads

TYPES OF DOCUMENTS

D = Techdata Sheets; **R** = Technical Reports and Technical Notes; **G** = NCEL Guides and Abstracts; **I** = Index to TDS; **U** = User Guides; ☐ None - remove my name

Old Address:

_____

_____

_____

_____

Telephone No.: _____

New Address:

_____

_____

_____

_____

Telephone No.: _____

INSTRUCTIONS

The Naval Civil Engineering Laboratory has revised its primary distribution lists. To help us verify our records and update our data base, please do the following:

- Add - circle number on list
- Remove my name from all your lists - check box on list.
- Change my address - add telephone number
- Number of copies should be entered after the title of the subject categories you select.
- Are we sending you the correct type of document? If not, circle the type(s) of document(s) you want to receive listed on the back of this card.

Fold on line, staple, and drop in mail.

DEPARTMENT OF THE NAVY

Naval Civil Engineering Laboratory
Port Hueneme, CA 93043-5003

Official Business
Penalty for Private Use, $300

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 12503 WASH D.C.

POSTAGE WILL BE PAID BY ADDRESSEE

CODE L34 (J LEDERER)
COMMANDING OFFICER
NAVAL CIVIL ENGINEERING LABORATORY
PORT HUENEME CA 93043-5003

NCEL DOCUMENT EVALUATION

You are number one with us; how do we rate with you?

We at NCEL want to provide you our customer the best possible reports but we need your help. Therefore, I ask you to please take the time from your busy schedule to fill out this questionnaire. Your response will assist us in providing the best reports possible for our users. I wish to thank you in advance for your assistance. I assure you that the information you provide will help us to be more responsive to your future needs.

R. N. STORER, Ph.D, P.E.
Technical Director

DOCUMENT NO. _____ TITLE OF DOCUMENT: _____

_____

Date: _____ Respondent Organization : _____

Name: _____ Activity Code: _____
Phone: _____ Grade/Rank: _____

Category (*please check*):

Sponsor ____ User ____ Proponent ____ Other (*Specify*) _____

Please answer on your behalf only; not on your organization's. Please check (*use an X*) only the block that most closely describes your attitude or feeling toward that statement:

SA Strongly Agree A Agree O Neutral D Disagree SD Strongly Disagree

---

	SA	A	N	D	SD
1. *The technical quality of the report* is comparable to most of my other sources of technical information.	( )	( )	( )	( )	( )
2. The report will make significant improvements in the cost and or performance of my operation.	( )	( )	( )	( )	( )
3. The report acknowledges related work accomplished by others.	( )	( )	( )	( )	( )
4. The report is well formatted.	( )	( )	( )	( )	( )
5. The report is clearly written.	( )	( )	( )	( )	( )
6. The conclusions and recommendations are clear and directly supported by the contents of the report.	( )	( )	( )	( )	( )
7. The graphics, tables, and photographs are well done.	( )	( )	( )	( )	( )

Do you wish to continue getting NCEL reports? ☐ YES ☐ NO

Please add any comments (e.g., in what ways can we improve the quality of our reports?) on the back of this form.

Comments:

Please fold on line and staple

DEPARTMENT OF THE NAVY

Naval Civil Engineering Laboratory
Port Hueneme, CA 93043-5003

Official Business
Penalty for Private Use $300

Code L03B
NAVAL CIVIL ENGINEERING LABORATORY
PORT HUENEME, CA 93043-5003